UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

Diseño de un PCB educativo de bajo coste para ESO, FP y Bachillerato

Autor: David Ramírez Mesón
Tutor: Juan Antonio Andrés Sáez
Ponente: Sergio López Buedo

junio 2019

David Ramírez Mesón
Diseño de un PCB educativo de bajo coste para ESO, FP y Bachillerato

David Ramírez Mesón

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

A mi bisabuela. Estés donde estés siempre te recordaré.

Agradecimientos

En primer lugar, me gustaría agradecer al Colegio Decroly la oportunidad que me han brindado al permitirme hacer este trabajo de fin de grado con ellos, otorgándome la responsabilidad de que los alumnos de secundaria y bachillerato utilizaran mi trabajo para aprender a programar. Me gustaría destacar la ayuda de Gonzalo Fernández de Tejada Garay, ya que ha sido el docente del colegio con el que he contactado todos estos meses.

También agradecerle a mi tutor, Juan Antonio Andrés Sáez, por haber depositado la confianza de realizar un cometido de esta envergadura y por haberme puesto en contacto con este colegio.

Cómo no, recordar a toda la gente que me ha apoyado y ayudado todos estos años y, sobre todo, mis amigos y familiares que siempre han estado dándome fuerzas para continuar estudiando una ingeniería tan complicada como ésta, pero haciéndome ver que todo esfuerzo conlleva una gran recompensa y sintiéndose orgullosos de mí por cada objetivo que lograba.

Para concluir, me gustaría hacer una mención especial a mis padres, porque sin ellos no habría logrado llegar a ser quien soy hoy en día, y a mi hermana mayor, Judit, que es una gran ingeniera que me ha enseñado que rendirse no es una opción.

Resumen

Este Trabajo Fin de Grado consiste en la creación de una placa de circuito impreso, PCB por sus siglas en inglés, de bajo coste que sirva como placa portadora (HAT) para varios sensores para poder impartir clases de robótica a nivel de ESO, Bachillerato y FP. Este PCB tiene la característica de poder conectarse mediante el puerto de 40 pines que contiene tanto la Raspberry Pi (RPi) [26] como su alternativa de bajo coste Orange Pi (OPi) [20], además de cualquier ordenador que tenga incluido este puerto. De esta forma puede aprenderse a programar de un modo dinámico. La idea es que todo lo que se programe se pueda visualizar dependiendo del sensor que esté conectado en ese momento, pudiendo hacer desde algo tan sencillo como encender un LED hasta algo más complejo como hacer un termómetro.

También se ha desarrollado todo el software necesario para el funcionamiento de todos los sensores en el lenguaje Python [25] para facilitar la labor del docente, así como un complemento para poder utilizar esta los 40 pines del GPIO en Scratch [30]. Scratch es un entorno de programación con fines didácticos para aprender a programar de una forma visual directamente seleccionando con el ratón las acciones deseadas, definidas a través de bloques, de forma intuitiva y sencilla.

El objetivo final es que crear un kit completo para aprender programación y robótica, que compita con los que ya existen de Arduino [3] o Mindstorms de Lego [13], pero que sea más barato, para que puedan permitírselo colegios e institutos que tengan recursos económicos más limitados, y que utilicen a su vez un mejor hardware, de tal forma que el importe total del kit, incluyendo la placa de desarrollo Orange Pi PC+, el (HAT) y un juego de 37 sensores, sea de un máximo de 50€.

Palabras clave

PCB, HAT, OPi, RPi, BPi, Orange Pi, Raspberry Pi, Banana Pi, Bajo coste, Aprender, Robótica, Colegio, Instituto, Diseño, Programación

Abstract

This Final Degree Project consists on the creation of a low cost Printed Circuit Board (PCB) used as a Hardware Attached on Top (HAT) for several sensors in order to teach robotics at the educational level of secondary and baccalaureate. This PCB has the feature of being able to connect to the 40-pin port that contains the Raspberry Pi [26] and its low-cost alternative Orange Pi [20], in addition to any computer that has this port included. In this way you can learn to program in a dynamic way. The idea is that everything that is programmed can be visualized depending on the sensor that is connected at that moment, being able to do something as simple as turning on an LED to something more complex like making a thermometer.

It has also been developed in the Python language [25] all the software necessary to make all sensors operative to facilitate the work of the teacher, as well as a complement to be able to use the 40 pins of the GPIO in Scratch [30]. Scratch is a programming environment with didactic purposes to learn how to program in a visual way directly by selecting the desired actions with the mouse, defined through blocks, in an intuitive and simple way.

The ultimate goal is to create a complete kit to learn programming and robotics, which will compete with those that already exist like Arduino [3] or Lego Mindstorms [13], but which is cheaper, so that schools and high schools that have more limited economic resources can afford it, and at the same time using a better hardware, so that the total amount of the kit, including the Orange Pi PC+ development board, the HAT and a set of 37 sensors, is a maximum of €50.

Keywords

PCB, HAT, OPi, RPi, BPi, Orange Pi, Raspberry Pi, Low-cost, Learning, Robotics, School, High-school, Design, Programming

ÍNDICE

1 Introducción **1**

1.1 Motivación 1

1.2 Objetivos 2

1.3 Organización de la memoria 2

2 Estado del arte **5**

2.1 Single Board Computer y otras alternativas 5

2.2 Conclusión 6

3 Definición del sistema **7**

3.1 Metodología 7

3.2 Herramientas e instrumentación utilizadas 8

4 Diseño **11**

5 Desarrollo **15**

5.1 Python 15

5.2 Scratch 17

5.3 PIC18F4455 18

6 Presupuesto **19**

7 Experimentos realizados y resultados **21**

7.1 Experimentos realizados 21

7.1.1 Reuniones 21

7.1.2 Taller 22

7.2 Resultados 27

8 Conclusiones y trabajo futuro **33**

Bibliografía **36**

Definiciones **37**

Acrónimos **39**

Apéndices **41**

A Guía para utilizar la Orange Pi **43**

A.1 Instalación del sistema operativo Armbian en la Orange Pi 43

A.2 Preparación de la Orange Pi por primera vez 45

A.2.1 Orange Pi con un entorno de escritorio 45
A.2.2 Orange Pi como servidor. Conexión con PuTTY y SSH 45
A.3 Configuración en la Orange Pi 46
A.3.1 Configuración inicial 46
B Sensores **51**

Listas

Lista de códigos

5.1 Test sensores 16

7.1 Ejercicio 1 25
7.2 Ejercicio 2 26
7.3 Ejercicio 1 hecho durante el taller 27
7.4 Ejercicio 2 hecho durante el taller 28

Lista de figuras

3.1 Modelo de Ciclo de Vida Cascada con Subproyectos 7
3.2 Logo de Armbian 8
3.3 Logo de Altium 9
3.4 Logo de Scratch 9
3.5 Logo de MPLAB® X Integrated Development Environment (IDE) 9
3.6 PICkit™ 3 In-Circuit Debugger 9

4.1 Kit de 37 sensores 11
4.2 Pinout 12
4.3 Pinout 13
4.4 Primer encargo HAT 14
4.5 Segundo encargo HAT 14

5.1 Interfaces de comunicación SPI, I2C, UART 16
5.2 HAT con la pantalla OLED y sensores figura B.8 y figura B.37 17
5.3 Bloques creados para Scratch 3 17
5.4 Bloque de emisión de órdenes 17
5.5 18
5.6 Bloques para asignar una variable 18
5.7 Bloque para leer un valor sin asignar una variable 18
5.8 PIC18F4455 18

6.1 Tabla resumen del presupuesto 20

7.1 Esquema de los pines del HAT 23

7.2 Sensores conectados a la OPi por un grupo de estudiantes 23
7.3 Un grupo de estudiantes programando los sensores acoplados al HAT de la OPi 24
7.4 Orange Pi con diversos sensores acoplados 29
7.5 Grupo de estudiantes del Colegio Decroly durante el taller programando en Python . . . 30
7.6 Grupo de estudiantes del Colegio Decroly durante el taller consigue iluminar un LED . 31

A.1 Sistema operativo Armbian Bionic 43
A.2 Administrador de discos 44
A.3 Win32DiskImager 44
A.4 Esquema de conexión de la Orange Pi 45
A.5 Esquema de conexión de la Orange Pi 46
A.6 Menú el programa PuTTY 47
A.7 Menú principal de Armbian 48
A.8 Menú System de armbian-config 49

B.1 Sensor KY-001 51
B.2 Sensor KY-002 51
B.3 Sensor KY-003 52
B.4 Sensor KY-004 52
B.5 Sensor KY-005 52
B.6 Sensor KY-006 53
B.7 Sensor KY-008 53
B.8 Sensor KY-009 53
B.9 Sensor KY-010 54
B.10 Sensor KY-011 54
B.11 Sensor KY-012 54
B.12 Sensor KY-013 55
B.13 Sensor KY-015 55
B.14 Sensor KY-016 55
B.15 Sensor KY-017 56
B.16 Sensor KY-018 56
B.17 Sensor KY-019 56
B.18 Sensor KY-020 57
B.19 Sensor KY-021 57
B.20 Sensor KY-022 57
B.21 Sensor KY-023 58
B.22 Sensor KY-024 58
B.23 Sensor KY-025 58
B.24 Sensor KY-026 59

B.25 Sensor KY-027 59
B.26 Sensor KY-028 59
B.27 Sensor KY-029 60
B.28 Sensor KY-031 60
B.29 Sensor KY-032 60
B.30 Sensor KY-033 61
B.31 Sensor KY-034 61
B.32 Sensor KY-035 61
B.33 Sensor KY-036 62
B.34 Sensor KY-037 62
B.35 Sensor KY-038 62
B.36 Sensor KY-039 63
B.37 Sensor KY-040 63

1

Introducción

1.1. Motivación

Desde hace unos años, la sociedad avanza a pasos agigantados hacia un mundo donde la tecnología va a formar una parte imprescindible de nuestra vida. Por ello, las nuevas generaciones, necesitan nuevos métodos de aprendizaje que se adapten a las nuevas necesidades. La programación es uno de estos nuevos requerimientos porque es algo que está a la orden del día, ya que cualquier sistema tecnológico necesita ser programado: páginas web, móviles, coches, aviones, semáforos... Por ello la programación es una habilidad muy importante e imprescindible actualmente. La *programación* se describe como el proceso por el cual se organiza una secuencia de pasos ordenados a seguir para realizar una acción [9].

En algunos países, entre los que se encuentra España [10], ya se ha comenzado a enseñar a programar a niños y adolescentes. La programación ayuda a pensar lógica y estructuradamente, lo que es de gran ayuda en temas relacionados con las matemáticas y la robótica.

Además, tanto los niños como los adolescentes, están en una etapa en la que tienen una facilidad de aprendizaje mayor que en la edad adulta, por lo que hay que aprovechar esta situación para que aprendan de una forma fácil y divertida.

La motivación principal de este TFG es que los colegios e institutos con menos recursos tengan la posibilidad de impartir clases de programación y robótica, para que así, los estudiantes puedan aprender de una forma entretenida y lúdica.

Asimismo, otra de las motivaciones, es que el tipo de programación que se va a desarrollar es más visual. Se va a utilizar un kit de 37 sensores intercambiables que, o bien mediante el programa Scratch, o bien utilizando el lenguaje Python, podrán ser programados. De esta forma, hay una cantidad muy elevada de posibles combinaciones prácticas, que en función de los sensores utilizados aumentarán la complejidad.

Por consiguiente, este proyecto hace realidad todas estas motivaciones para facilitar la mejora y el progreso hacia un sistema educativo más tecnológico.

1.2. Objetivos

El objetivo principal de este proyecto es conseguir crear un sistema de bajo coste mediante el cual se pueda enseñar programación y robótica a nivel de la ESO, FP y Bachillerato. Se pretende hacer de una forma sencilla y entretenida, de forma que el método de aprendizaje motive a los estudiantes.

Este proyecto se compone por dos partes: una hardware y otra software. La primera, el hardware, consiste en un SBC que contiene un puerto GPIO de 40 pines. Se ha creado un HAT , módulo que se acopla en la parte superior de puerto para cumplir con los requisitos necesarios para facilitar las labores de aprendizaje. A su vez, se dispone de un kit de 37 sensores programables que se acoplan alrededor de este HAT. De esta forma, los estudiantes pueden hacer diversas combinaciones con los sensores y aprender a programarlos todos. Además, algunos sensores requieren de la utilización de protocolos específicos como el SPI e I2C [15], que otorga más posibilidades a la hora de profundizar en el tema del hardware. Por otro lado está el software, cuyo objetivo es que pueda aprenderse o bien utilizando la herramienta Scratch [30] para aprender a programar de una forma visual; o bien Python [25], donde se darán ejemplos funcionales donde progresivamente se vayan aprendiendo nuevos conceptos, desde los más sencillos hasta otros más complejos.

1.3. Organización de la memoria

La memoria de este Trabajo de Fin de Grado se organiza en nueve capítulos, de los cuales, el primero es el actual:

1.– Introducción
2.– Estado del arte
3.– Definición del sistema
4.– Diseño
5.– Desarrollo
6.– Presupuesto
7.– Reuniones
8.– Experimentos realizados y resultados
9.– Conclusiones y trabajo futuro

En la sección del *Estado del arte* se detalla qué son los SBCs, qué les caracteriza y qué otras alternativas existen para que los niños aprendan programación y robótica. También se explica cuáles son las desventajas de la utilización de otras alternativas frente a este proyecto.

En el capítulo *Definición del sistema* se define la metodología utilizada para el diseño y desarrollo del proyecto, así como un desglose de los pasos acontecidos dentro de cada ciclo de esta metodología. También se detallan las herramientas e instrumentación con la que se ha diseñado y desarrollado el

proyecto.

En el siguiente capítulo, *Diseño*, se explica con detalle cómo se ha diseñado el HAT y qué requerimientos se han tenido en cuenta para cumplir con los objetivos establecidos.

El apartado *Desarrollo* describe cómo se ha implementado este proyecto. También se especifica el progreso llevado a cabo para desarrollar y testear la funcionalidad de cada recurso creado, y un ejemplo del código utilizado para algunos sensores.

En el capítulo *Presupuesto* se desglosan los costes reales para la fabricación de las 30 placas solicitadas por el cliente, además de todos los componentes necesarios.

La sección *Experimentos realizados y resultados* es una de las más importantes, ya que especifica todos los factores que se han tenido en cuenta para optimizar los costes y las capacidades de la placa. Además cuenta con un registro de las reuniones realizadas con el cliente y los resultados obtenidos después de realizar todos los experimentos.

En las *Conclusiones y trabajo futuro* se hace una valoración general del trabajo realizado y se analizan posibles mejoras en el futuro.

Por último, este documento incluye en sus páginas iniciales un índice general y otro de figuras. Del mismo modo, en las últimas páginas incluye la bibliografía con todas las referencias utilizadas, un anexo de definiciones, otro de acrónimos y un apéndice con la *Guía para utilizar la Orange Pi* y los *Sensores*.

2

Estado del arte

Con el auge tecnológico de los últimos años, han incrementado las iniciativas para mejorar los métodos actuales de aprendizaje en las generaciones venideras. Especialmente en los colegios e institutos se está intentando impulsar métodos más innovadores utilizando la tecnología, mediante los cuales los alumnos desarrollen diferentes habilidades y capacidades adaptadas al mundo actual que con los métodos tradicionales de enseñanza no lograrían.

En esta sección se describe qué son los SBC y qué alternativas hay a la propuesta y defendida en este proyecto.

2.1. Single Board Computer y otras alternativas

Los Single Board Computer (SBC) u ordenadores de placa reducida en español, son computadores con una serie de características distintas a los PCs convencionales [11].

La primera peculiaridad de los SBCs es que tiene unas dimensiones muy reducidas. En el caso de los SBCs utilizados en el desarrollo de este proyecto, la OPi y la RPi, tienen unas dimensiones de 8,5 x 5,3 cm, que es un tamaño similar al de una tarjeta de crédito.

Otro rango distintivo es su bajo coste, ya que se pueden adquirir por menos de 10 €, como pueden ser la OPi Zero [21] o la RPi Zero [27], y además no superan los 100 €, salvo alguna excepción, como Coral [8], la nueva SBC creada por Google que cuesta 149,99 $.

Por último, otra particularidad de estas placas es que son de bajo consumo y trabajan a poca potencia, lo que permite utilizarlas como servidor multimedia o para ofimática.

Orange Pi, Raspberry Pi y Banana Pi

El HAT desarrollado para este proyecto se ha hecho pensando en que hoy en día se está estandarizando la utilización de un puerto de 40 pines entre distintas compañías fabricantes de SBCs de forma que sea posible su utilización independientemente de la placa que se adquiera. Este tema se abordará con más detalle en el **capítulo 4**.

Arduino

Arduino es una plataforma electrónica de bajo coste de código abierto. Las placas Arduino tienen múltiples funcionalidades, transformando señales de entrada leídas a través de un sensor, en señales de salida enviadas a un sensor distinto. Por ejemplo leer la entrada de un sensor de luminosidad y hacer que se encienda un led. Para hacerlo, utiliza el lenguaje de programación Arduino y el software Arduino (IDE) [4].

Los principales motivos por los que se ha decidido utilizar una alternativa a Arduino para este proyecto son el uso de un lenguaje de programación propio, siendo necesario instalar su propio software IDE, y que el HAT no podría utilizarse en otros SBCs.

Lego Mindstorms

Lego Mindstorms es una línea de juguetes de robótica para niños con la que pueden aprender a programar utilizando bloques con diferentes funciones y construir sus propios robots. El principal problema de esta tecnología es su coste, puesto que cuesta 399,99 € [13] y es algo que no se pueden permitir todos los centros educativos.

2.2. Conclusión

Para concluir el estado del arte, puede apreciarse que en contexto actual existen varias alternativas cuyos objetivos son similares al propuesto en este documento, con la diferencia de que en este proyecto se han aunado todas las ventajas, además de añadir características nuevas.

Las propuestas actuales, o bien son de bajo coste y muy complejas para niños pequeños, o bien son sencillas de utilizar pero y muy caras.

Estas características son las que se han tomado como referencia para desarrollar este TFG, ya que se ha utilizado una placa de desarrollo de bajo coste en la que se puede hacer programación mediante el programa Scratch que contiene bloques, además de poder programar en Python, que es un lenguaje muy utilizado actualmente debido a su sencillez y potencial. Por último, si se quiere aprender a programar a bajo nivel se puede hacer en el lenguaje de programación C.

3

Definición del sistema

3.1. Metodología

Para este proyecto se ha utilizado una metodología de Ciclo de Vida en Cascada Incremental. Se ha elegido una metodología tradicional ya que tenía varias vías de desarrollo que podían producirse en paralelo [23].

Se propusieron varios objetivos independientes, que son explicados con detalle a lo largo del documento, haciendo análisis, diseño, codificación y prueba de cada parte del proyecto, para finalmente unificarlo todo. De esta forma, el cliente podía ver poco a poco el progreso llevado a cabo en cada área, pudiendo proponer mejoras después de cada reunión concertada. Mediante esta metodología no fue necesario quedar con demasiada regularidad, sino que cuando se terminaba la funcionalidad de una de las partes se concertaba una cita y se mostraba esa funcionalidad desarrollada. También se aprovechaba para explicar cómo se estaban abordando en paralelo el resto de funcionalidades, con la ventaja de que, al haber un único desarrollador, se podía escoger qué parte desarrollar cada día, sin afectar al resto de funcionalidades.

La **figura 3.1** es un ejemplo gráfico del tipo de metodología utilizada, donde puede verse las fases que se han seguido para todo el proceso.

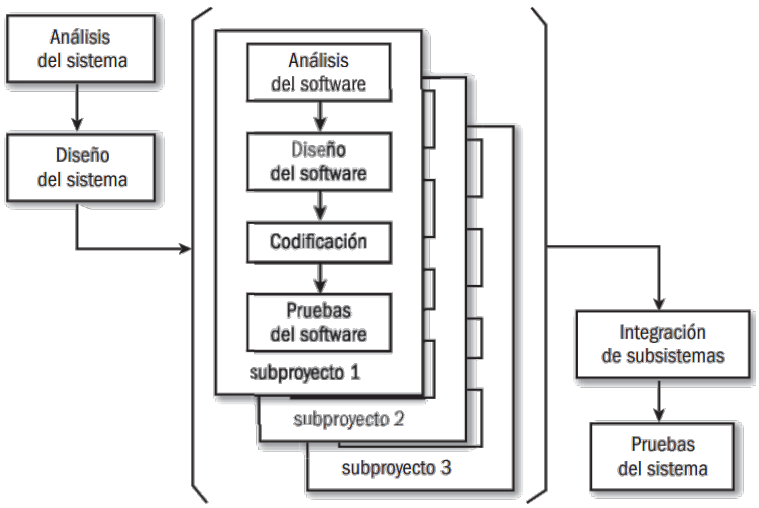


Figura 3.1: Modelo de Ciclo de Vida Cascada con Subproyectos

Análisis del sistema

1.– Planificación del estado del arte.

2.– Estudio de alternativas de bajo coste.

3.– Especificación de objetivos y funcionalidad.

Diseño del sistema

1.– Diseño del hardware.

2.– Diseño del software con Python.

3.– Diseño del complemento para Scratch.

Codificación

1.– Comunicación de los pines del SBC con los sensores.

2.– Implementación de las funciones para cada sensor digital.

3.– Implementación de la función ADC en el PIC18F4455.

4.– Implementación de las funciones para cada sensor analógico.

5.– Comunicación de los pines del PIC18F4455 con los de los sensores.

Pruebas

1.– Planificación de las pruebas.

2.– Integración de los subsistemas.

3.– Desarrollo de las pruebas.

4.– Recopilación de datos.

5.– Análisis de resultados.

6.– Evaluación del sistema desarrollado.

3.2. Herramientas e instrumentación utilizadas

En este apartado se describen las herramientas utilizadas para el desarrollo de este proyecto, así como la instrumentación con la que se ha comprobado que los datos recopilados eran los esperados.

Armbian

Armbian es un sistema operativo para SBCs basado en Debian y Ubuntu para procesadores ARM [5]. Este sistema operativo es en el que se ha instalado y desarrollado todo lo necesario para que el HAT funcione correctamente en la Orange Pi.

armbian

Figura 3.2: Logo de Armbian

Altium Designer

Figura 3.3: Logo de Altium

Altium Designer es programa de diseño de PCB para placas de circuito impreso. Se ha utilizado para la creación del HAT que se puede acoplar a los distintos SBCs. Con este programa se crearon los gerbers necesarios para la fabricación de la placa en China [2].

Scratch

Scratch es un lenguaje de programación gratuito y una comunidad en línea donde puedes crear tus propias historias, juegos y animaciones interactivas [30]. Se ha creado un complemento que se comunica con el HAT, de tal forma que se puede interactuar con los diferentes sensores de una forma visual y sencilla.

Figura 3.4: Logo de Scratch

MPLAB® X IDE

Figura 3.5: Logo de MPLAB® X Integrated Development Environment (IDE)

MPLAB es un entorno de desarrollo integrado gratuito IDE, destinado a productos de la marca Microchip. Este editor es modular, permite seleccionar los distintos microcontroladores soportados, además de permitir la grabación de estos circuitos integrados directamente al programador [19].

PICkit™ 3 In-Circuit Debugger

El MPLAB® PICkit ™ 3 In-Circuit Debugger / Programmer permite la depuración y la programación rápidas y sencillas de los microcontroladores PIC® utilizando la potente interfaz gráfica de usuario del entorno de desarrollo integrado MPLAB® X IDE [18]. Se ha utilizado para programar el PIC18F4455 del HAT.

Figura 3.6: PICkit™ 3 In-Circuit Debugger

4

Diseño

Para el diseño de este proyecto se ha tenido en cuenta lo analizado en el estado del arte, además de las recomendaciones y posibles mejoras propuestas por profesionales de la enseñanza y del diseño de hardware.

En un principio, se decidió implementar una placa estática, con todos los sensores programables integrados. Debido a la motivación de este proyecto, finalmente se ha utilizado un kit de 37 sensores de bajo coste, como el de la **figura 4.1**

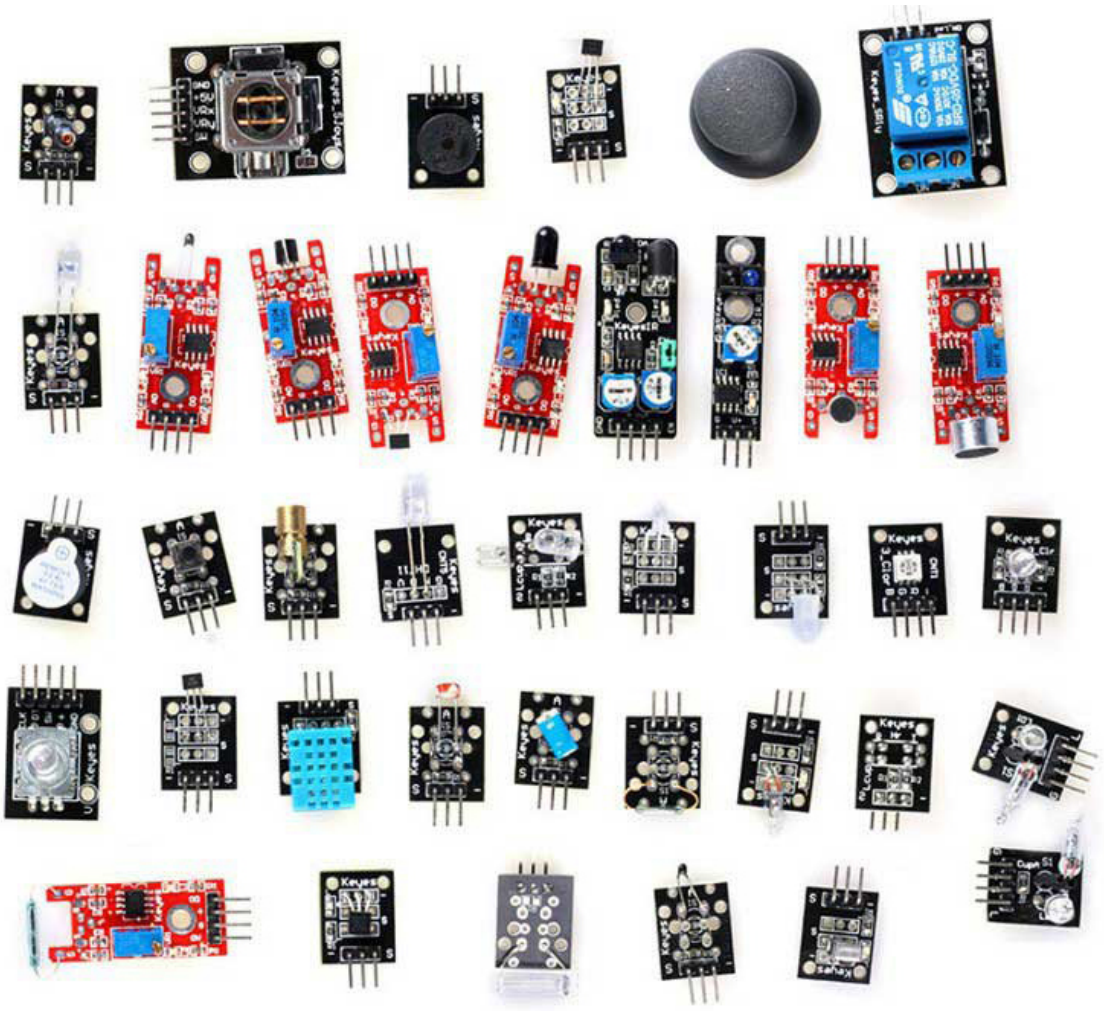
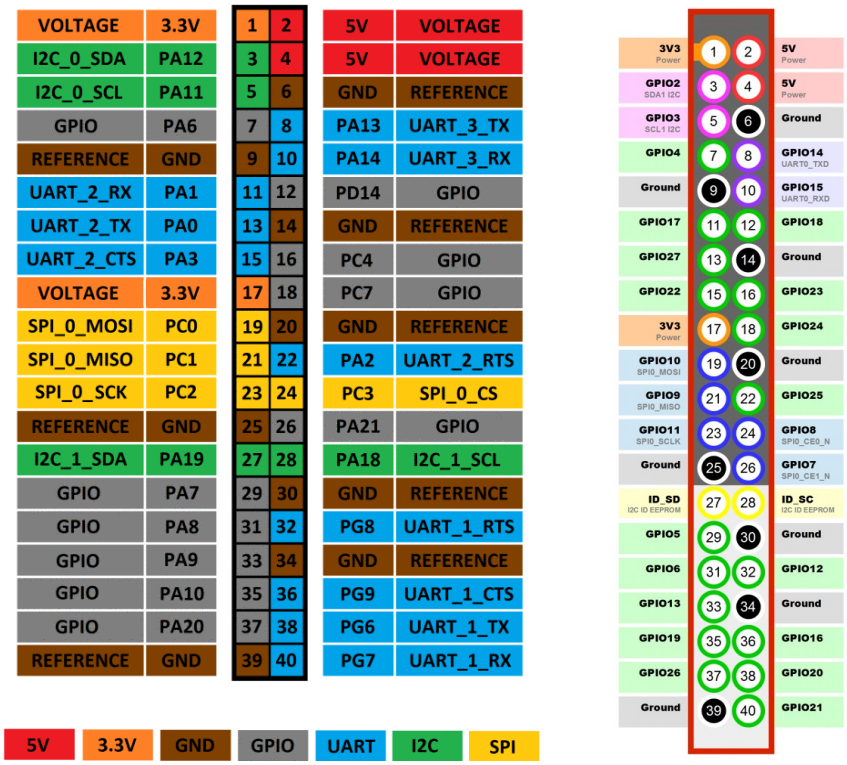


Figura 4.1: Kit de 37 sensores

La mayor ventaja de la utilización de este kit es que al haber tantos sensores pueden utilizarse en varios cursos, incrementando la dificultad gradualmente en función del sensor que se quiera programar.

Uno de los objetivos más importantes durante el diseño era hacer algo que no solo fuera didáctico, sino también divertido para los estudiantes. A esto se le suman los beneficios que aporta tener nociones en programación desde niño, debido a que te ayuda a pensar de una manera estructurada y lógica y puede repercutir positivamente a largo plazo [1]. Este es el principal motivo por el que se ha comenzado a impartir este tipo de enseñanzas en varios lugares de Europa, entre los que se encuentra España [16], pero donde todavía no se ha estandarizado en todos los centros educativos.



(a) Orange Pi pinout

(b) Raspberry Pi y Banana Pi pinout

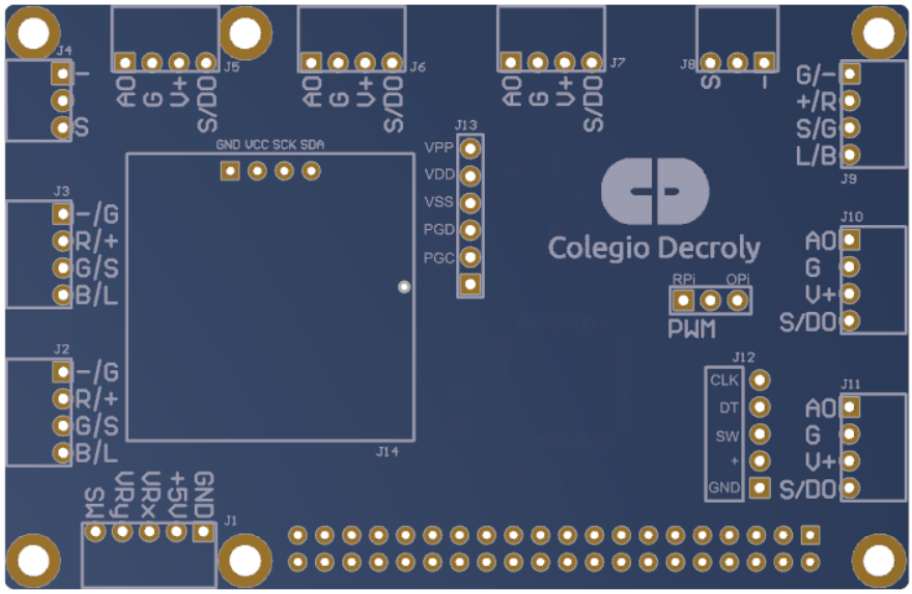
Figura 4.2: Pinout de los SBCs para los que se ha diseñado el HAT

Se diseñó un HAT acoplable al puerto de 40 pines de la Orange Pi [20] como el de la **figura 4.2(a)**, así como de la Raspberry Pi [26] y Banana Pi [6] **figura 4.2(b)**. Se tuvo en cuenta que todos los pines coincidían, excepto el del Pulse-Width Modulation (PWM) , por lo que se creó un jumper mediante el cual se puede elegir si se está utilizando la OPi, cuyo pin es el 7, o la RPi y BPi, cuyo pin es el 32.

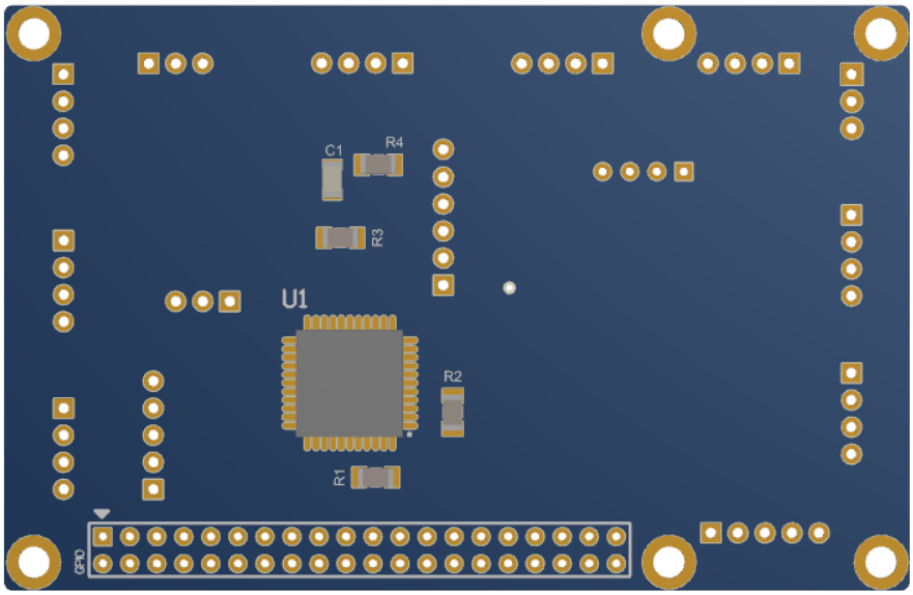
También se tuvieron en cuenta los distintos tipos de sensores, de forma que todos ellos pudieran acoplarse en alguna de las cabeceras distribuidas alrededor del HAT, y teniendo en cuenta que pudieran ponerse el mayor número posible de sensores a la vez.

Finalmente, el diseño del HAT quedó como en la **figura 4.3**. Se implementó utilizando la herramienta

previamente mencionada, llamada Altium Designer. Posteriormente, se generaron los archivos gerber, que contienen la información necesaria para la fabricación del PCB, y fueron enviados a una empresa especializada para su elaboración.



(a) Cara superior



(b) Cara posterior

Figura 4.3: Diseño de HAT para la OPi, RPi y BPi

Fue necesario hacer dos encargos, ya que el primero fue una versión de prueba para encontrar posibles errores, por lo que se solicitaron solo 5 PCBs, como los de la **figura 4.4** a la empresa PCBWay [22]. El segundo encargo fue la versión final y se encargó a la empresa JLCPCB [12]. Para este pedido se tuvo en cuenta que la fabricación tenía que cumplir con el directiva RoHS [29] y se subsanaron algunos pequeños errores. En la **figura 4.5** se puede ver el resultado final de este segundo HAT.

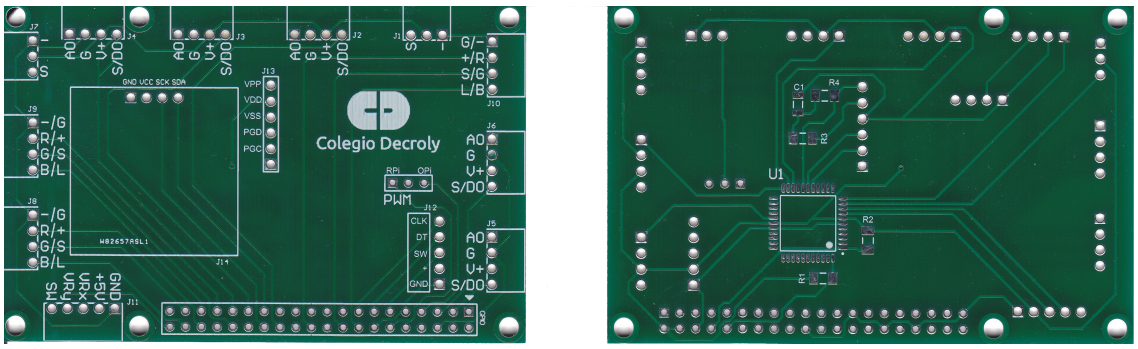


Figura 4.4: Primer encargo HAT

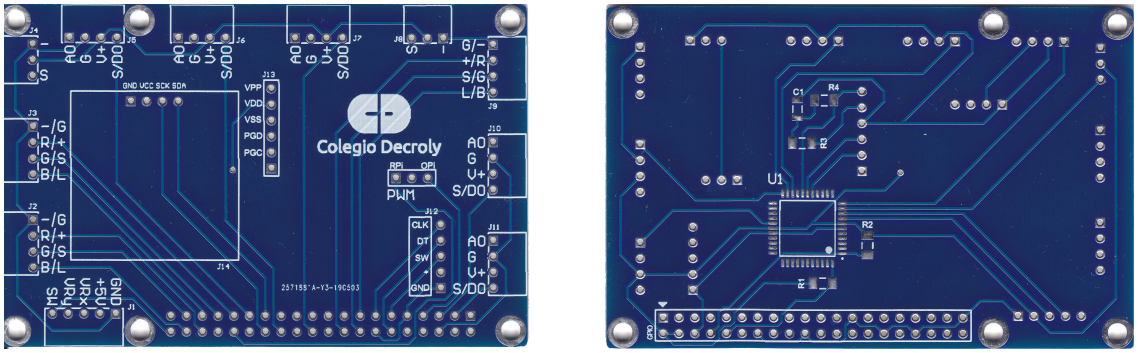


Figura 4.5: Segundo encargo HAT

5

Desarrollo

En esta sección se habla con detalle sobre cómo partiendo de la base establecida en el diseño, se ha desarrollado este proyecto hasta llegar a la versión final que una vez testada cumple los objetivos marcados a lo largo de este documento.

Lo primero que se hizo fue instalar el sistema operativo Armbian [5], mencionado anteriormente, en una tarjeta micro SD y se prosiguió configurando e instalando todo lo necesario, como viene explicado en el **apéndice A**, para comenzar a desarrollar todo el software necesario.

5.1. Python

Se comenzó implementando en Python [25] el software necesario para comprobar que el sensor LED SMD RGB como el de la **figura B.8** funcionaba correctamente en todos los pines programables de la OPi. Posteriormente se procedió a desarrollar varios archivos para comprobar que todos los sensores mostrados en la **figura 4.1** funcionaban correctamente y se comunicaban con los pines digitales del GPIO de la OPi. En el **código 5.1** se muestra un ejemplo de un programa con el que se pueden probar varios sensores. Una vez que se habían programado todos los sensores digitales se procedió a decidir qué pines eran más convenientes de utilizar, ya que como se puede ver en la **figura 4.2**, los pines digitales también pueden activarse para utilizar los protocolos UART, SPI o I2C [15]. Se decidió reservar los pines correspondientes a estas interfaces para poderlas utilizar para otras funcionalidades y sensores. En la **figura 5.1** se muestran gráficamente cómo funcionan estos tres protocolos.

Universal Asynchronous Receiver-Transmitter (UART)

El protocolo UART, que se corresponde con los pines 8 y 10, decidió utilizarse para comunicarse con el PIC18F4455 [17]. De esta forma se puede programar este microcontrolador para transformar algunos pines digitales en otros analógicos utilizando los ADCs internos que tiene.

Inter-Integrated Circuit (I2C)

La interfaz I2C, que utiliza los pines 27 y 28, se reservaron para conectar una pantalla OLED de 0,96” [14] al HAT, como en la **figura 5.2**, que junto al sensor de la **figura B.37** permite moverse por las

diversas secciones de un menú con el que se puede ver información sobre la OPi. En este menú se muestran por ejemplo: la memoria virtual, memoria interna, dirección IP, etc.

Serial Peripheral Interface (SPI)

Los pines 19, 21, 23 y 24 correspondientes al protocolo SPI se dejaron libres para posibles funcionalidades que no han sido requeridas para este proyecto. Pese a que existe la pantalla OLED de 0,96" que funciona con este protocolo, en vez de con I2C, se decidió no utilizarla porque usa un número mayor de pines, pese a que el funcionamiento es similar.

Código 5.1: Código en Python para testear varios sensores digitales

```
1  import OPi.GPIO as GPIO
2  from time import sleep
3
4  GPIO.setboard(GPIO.PCPCPLUS) # Orange Pi PC board
5  GPIO.setmode(GPIO.BOARD)
6
7  # The input pin of the Sensor will be declared. Additional to that the pullup resistor will be activated.
8  GPIO_PIN = 7
9  GPIO.setup(GPIO_PIN, GPIO.IN, pull_up_down = GPIO.PUD_UP)
10
11 print "Sensor-Test [press ctrl+c to end it]"
12
13 # This output function will be started at signal detection
14 def outFunction(null):
15         print("Signal detected")
16
17 # At the moment of detecting a Signal ( falling signal edge ) the output function will be activated.
18 GPIO.add_event_detect(GPIO_PIN, GPIO.FALLING, callback=outFunction, bouncetime=100)
```

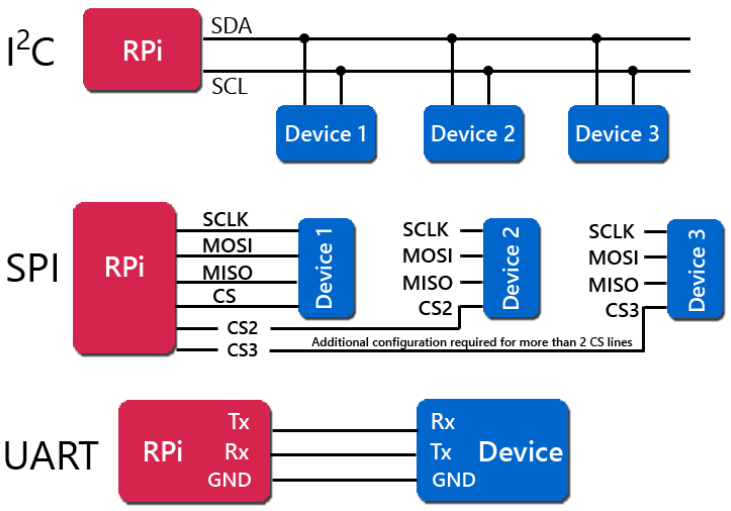


Figura 5.1: Interfaces de comunicación SPI, I2C, UART

Figura 5.2: HAT con la pantalla OLED y los sensores de la figura B.8 y B.37 funcionando

5.2. Scratch

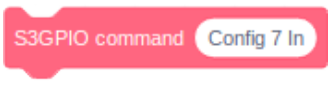
Además de lo implementado en el apartado anterior, se tuvo en cuenta que para empezar a programar es complicado ver un código tan complicado directamente. Por ello, se implementaron varios bloques, como los de la **figura 5.3**, para Scratch 3 [30] que hacen posible comunicar este programa con los pines de la OPi. Este software está adaptado de uno existente desarrollado para RPi [31].

Figura 5.3: Bloques creados para Scratch 3

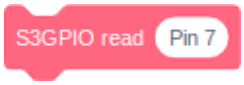
El bloque de la **figura 5.4** sirve para emitir a los pines la orden correspondiente. En el caso de la imagen, transmite al pin 11 la orden *on* para que se active. Si por ejemplo hubiera un LED conectado a dicho pin, se encendería. Para apagarlo habría que introducir *off* en lugar de *on*.

Figura 5.4: Bloque de emisión de órdenes

Por defecto, todos los pines están asignados como salidas, por lo que si se quieren configurar como entradas hay que usar el bloque de la **figura 5.5**. Una vez configurado como entrada puede conectarse un sensor, por ejemplo de movimiento, y leer su valor utilizando el bloque de la **figura 5.5(b)**.



(a) Bloque para asignar pin como entrada



(b) Bloque para leer pines asignados como entradas

Figura 5.5:

En la **figura 5.6** se muestra un ejemplo para asignar una variable llamada sensor al pin 7 que ha sido leído.

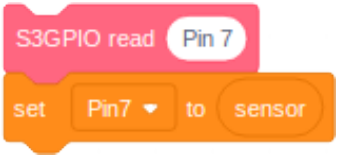


Figura 5.6: Bloques para asignar una variable al sensor que va a ser leído

Por último, si lo que se pretende es leer el valor de un sensor sin utilizar una variable, se puede hacer como se muestra en la **figura 5.7**

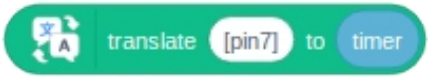


Figura 5.7: Bloque para leer un valor sin asignar una variable

5.3. PIC18F4455

El PIC18F4455 es el microcontrolador elegido para que el HAT pueda utilizar sensores analógicos, transformando varias señales digitales con los ADCs [17]. Para desarrollar el software necesario se utilizó el programa **MPLAB® X IDE** y para programarlo el **PICkit™ 3 In-Circuit Debugger**, ambos descritos en el **apartado 3.2**.

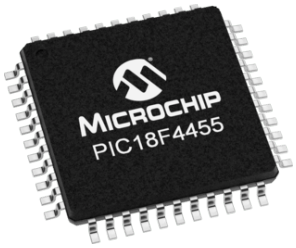


Figura 5.8: PIC18F4455

6

Presupuesto

Para llevar a cabo este proyecto, uno de los factores más importantes ha sido realizar un presupuesto que se ajuste a las necesidades del cliente. Para ello, se ha creado este presupuesto en el que se detallan los costes reales de un lote de 30 unidades, que es la cantidad solicitada por el Colegio Decroly.

Cada lote se compone de:

- Orange Pi PC+
- 37 sensores
- Pantalla OLED de 0,96"
- Adaptador VGA a HDMI
- HAT

Concepto	Coste Unitario	Cantidad	Cantidad Total
Materiales			
Orange Pi			
PC Plus	22,99 €	30	458,00 €
Gastos de envío			67,82 €
Subtotal			**757,72 €**
37 sensores			
Sensores	8,14 €	30	244,20 €
Gastos de envío			0 €
Subtotal			**244,20 €**
Pantalla Oled			
SSD1306	1,67 €	30	50,10 €
Gastos de envío			5,30 €
Subtotal			**55,40 €**
Adaptador			
VGA-HDMI	0,86 €	30	25,80 €
Gastos de envío			13,52 €

Concepto	Coste Unitario	Cantidad	Cantidad Total
Subtotal			**39,32 €**
HAT			
Fabricación	0,579 €	30	17,38 €
Componentes	5,02 €	30	150,50 €
Gastos de envío			10,80 €
Subtotal			**178,68 €**
Total			**1.275,32 €**

Para el lote solicitado de 30 unidades, el coste unitario es 42,51€ (impuestos no incluidos). En este presupuesto no se ha presupuestado la mano de obra por el montaje de los 30 PCBs, ya que al ser pocos no se ha contratado el servicio de montaje.

Description	30	
	Coste unitario	Total
Fabricación y montaje	5,96 €	178,68 €
Orange Pi PC+	25,26 €	757,72 €
SSD1306 I2C	1,85 €	55,40 €
Adaptador VGA-HDMI	1,31 €	39,32 €
Kit 37 sensores	8,14 €	244,20 €
TOTAL	42,51 €	**1.275,32 €**

Figura 6.1: Tabla resumen del presupuesto

7

Experimentos realizados y resultados

7.1. Experimentos realizados

Esta sección habla sobre los experimentos que se han realizado en un entorno real, centrándose en las reuniones con el cliente y en el taller con los estudiantes.

7.1.1. Reuniones

Previamente a la realización de los experimentos en un entorno real se realizaron una serie de reuniones con los representantes del Colegio Decroly [7] para poner en contexto los objetivos de éste proyecto y hacer un seguimiento del progreso que se estaba llevando hasta su versión final.

13/2/2018

La primera reunión fue para conocer a Gonzalo Fernández de Tejada Garay, el profesor de tecnología del Colegio Decroly. Él fue la persona de contacto dentro del centro durante todo el proceso.

Durante esa reunión se expuso la idea inicial del proyecto y cómo pensaba abordarse. A su vez, Gonzalo manifestó las necesidades del colegio para poder hacer un análisis exhaustivo de los requerimientos y vías de desarrollo para realizar el proyecto.

24/10/2018

La segunda cita tuvo lugar varios meses después y en ella se le mostró a Gonzalo un primer diseño del HAT en Altium. También vió la Orange Pi PC+ y se le explicaron las razones por las que se había elegido este SBC frente a otros más conocidos como la Raspberry Pi 3 Model B+.

También se habló del software, pues actualmente los alumnos del colegio aprenden a programar con Scratch durante la ESO y Arduino en Bachillerato. La conclusión de esa reunión fue que se tenía que poder enseñar a programar en Python, pero también en Scratch utilizando ese HAT.

6/03/2019

Después de la segunda reunión pasaron varios meses en los se mantuvo el contacto por correo electrónico para informar de los progresos. En la tercera, se había desarrollado el software para los sensores y también se había fabricado y testeado el primer encargo de PCBs, en los que se encontraron algunos pequeños errores que fueron subsanados en el segundo encargo.

Esta reunión, además de ser con Gonzalo, fue con Samuel Muñoz Martínez, el subdirector del colegio. Ese día pudieron ver el kit completo, con el HAT que tenía el logotipo del colegio, la pantalla OLED colocada en su lugar y la colocación de los sensores alrededor de la placa.

Ese día se concertó que a finales de mayo se realizaría un taller con una clase de 4º de ESO en la que se probaría la versión final del proyecto.

7.1.2. Taller

El día 31 de mayo de 2019 este proyecto fue probado en una clase de 4º de la ESO del Colegio Decroly [7], localizado en Madrid. Ese día además de Gonzalo y Samuel, también acudió Pablo Rodríguez Palenzuela, el dueño de ese colegio concertado, que estuvo presente durante el taller.

El taller estuvo constituido por 20 estudiantes, a los que se dividió en 4 grupos. Estos estudiantes no tenían conocimientos previos en programación en Python, por lo que era una buena forma para comprobar la efectividad del proyecto.

A cada equipo se le entregó una OPi con el HAT incorporado, un cable de alimentación, una bolsa con los componentes que tenían que programar y una hoja con un esquema de los pines del HAT como la de la **figura 7.1**.

Utilizando dicho esquema se les explicó dónde conectar cada componente, ya que había varias posibilidades para cada sensor. Por ejemplo, si conectaban el sensor en la cabecera J3 del esquema, podían utilizar los pines 31, 33 y 35. Además, tenían que hacer coincidir la serigrafía del esquema con la de los sensores, es decir, si se utiliza el sensor **figura B.14** que tiene como serigrafía -, R, G y B, las posibles posiciones son J2, J3 y J9.

Una vez se hubo explicado todo se propusieron dos ejercicios en Python: El primero para que aprendieran a utilizar los pines (**código 7.3**), y el segundo para hacer un programa más complejo en el que los estudiantes tenían que programar los sensores de la **figura B.14** y **B.4**, como se puede ver en el **código 7.4**.

La **figura 7.2** muestra cómo conectó uno de los grupos de estudiantes los sensores que se les otorgó, mientras que en la **figura 7.3** se puede ver como otro de los grupos está programando los sensores una vez conectados.

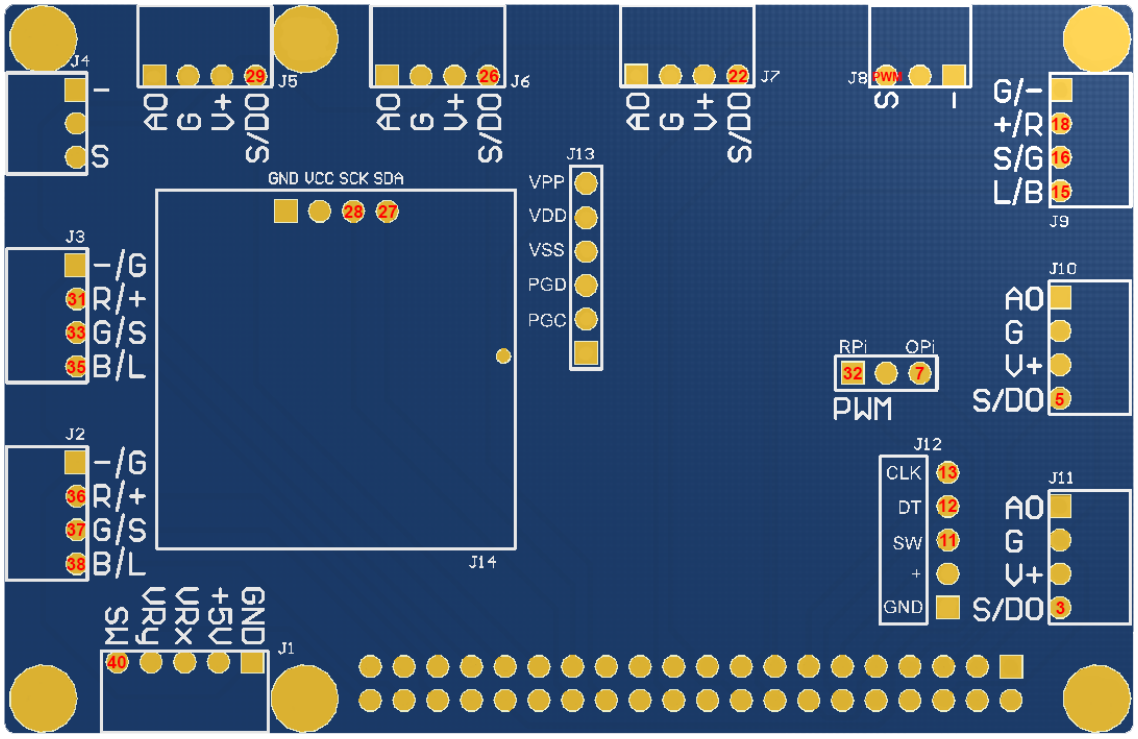


Figura 7.1: Esquema de los pines del HAT

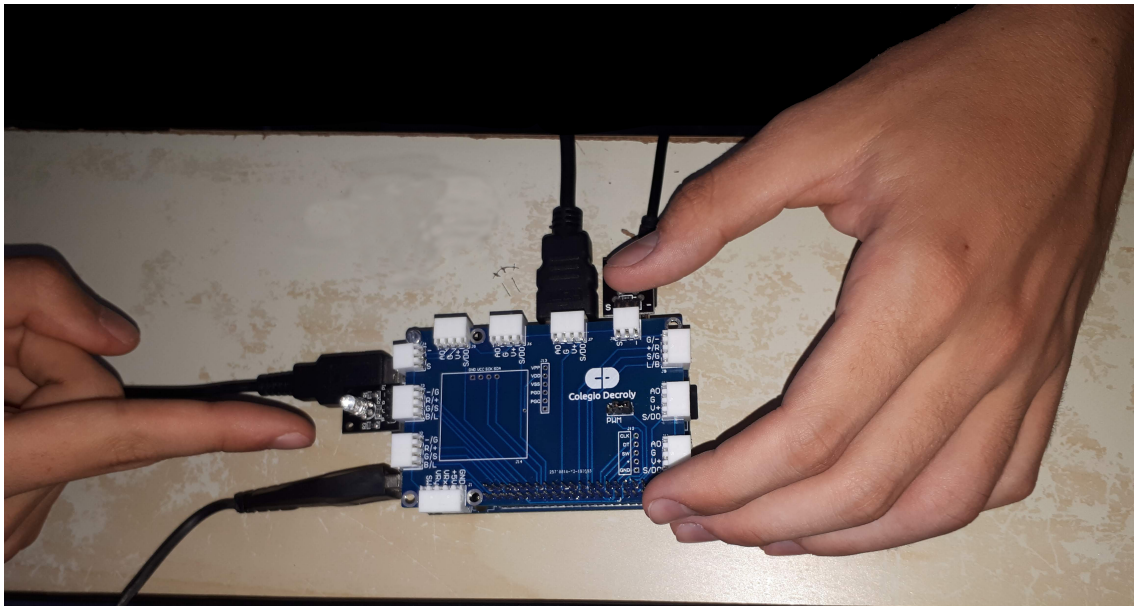


Figura 7.2: Sensores conectados a la Orange Pi por un grupo de estudiantes del Colegio Decroly durante el taller

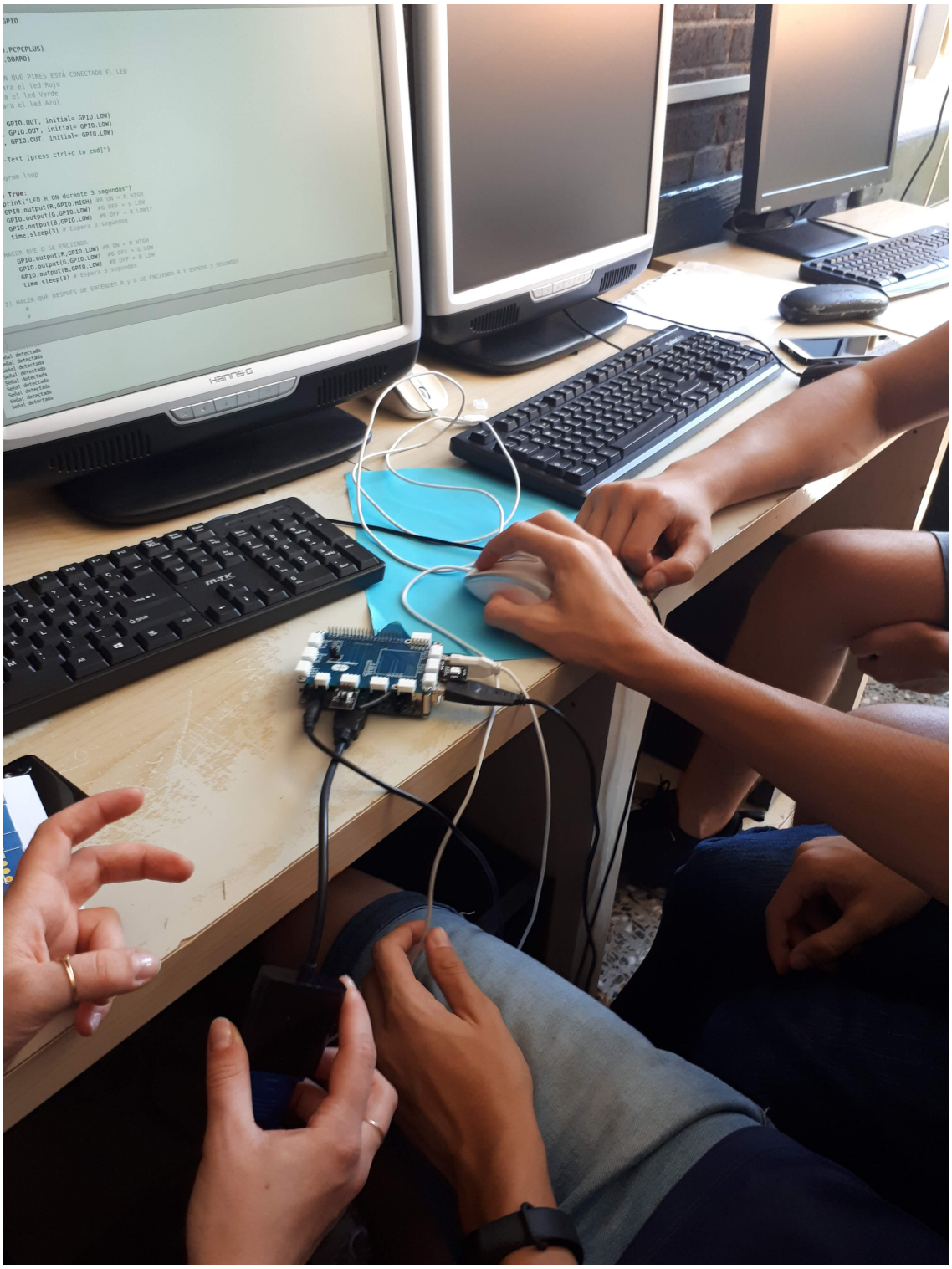


Figura 7.3: Un grupo de estudiantes programando los sensores acoplados al HAT de la OPi durante el taller

Código 7.1: Código del ejercicio 1 sin hacer

```
1  import OPi.GPIO as GPIO
2  import time
3
4  GPIO.setboard(GPIO.PCPCPLUS)
5  GPIO.setmode(GPIO.BOARD)
6
7  # DECLARAR EL PIN QUE VA A SER ACTIVADO
8  S = 37
9  GPIO.setup(S, GPIO.IN, pull_up_down = GPIO.PUD_UP)
10
11 print ("Compruebe el sensor")
12
13 # Esta función comienza cuando se detecta la señal
14 def funcion(null):
15     print("Señal detectada")
16
17 # Cuando se detecta una señal la salida de la función será activada
18 GPIO.add_event_detect(S, GPIO.FALLING, callback=funcion, bouncetime=100)
19
20 # Bucle principal
21 try:
22     while True:
23         time.sleep(1)
24
25 # Fin del programa
26 except KeyboardInterrupt:
27         GPIO.cleanup()
```

Código 7.2: Código del ejercicio 2 sin hacer

```
import OPi.GPIO as GPIO
import time

GPIO.setboard(GPIO.PCPCPLUS)
GPIO.setmode(GPIO.BOARD)

# 1) COMPROBAR EN QUÉ PINES ESTÁ CONECTADO EL LED
R = 18 #PIN para el led Rojo
G = 33 #PIN para el led Verde
B = 38 #PIN para el led Azul

GPIO.setup(R, GPIO.OUT, initial= GPIO.LOW)
GPIO.setup(G, GPIO.OUT, initial= GPIO.LOW)
GPIO.setup(B, GPIO.OUT, initial= GPIO.LOW)

print ("LED-Test [press ctrl+c to end]")

# Bucle principal
try:
    while True:
        print("LED R ON durante 3 segundos")
        GPIO.output(R,GPIO.HIGH) #R ON = R HIGH
        GPIO.output(G,GPIO.LOW) #G OFF = G LOW
        GPIO.output(B,GPIO.LOW) #B OFF = B LOWtr
        time.sleep(3) # Espera 3 segundos

# 2) HACER QUE G SE ENCIENDA
        GPIO.output(R,GPIO.LOW) #R ON = R HIGH
        GPIO.output(G,GPIO.LOW) #G OFF = G LOW
        GPIO.output(B,GPIO.LOW) #B OFF = B LOW
        time.sleep(3) # Espera 3 segundos

# 3) HACER QUE DESPUÉS DE ENCENDER R y G SE ENCIENDA B Y ESPERE 3 SEGUNDOS
        #
        #
        #
        #

# Fin del programa
except KeyboardInterrupt:
        GPIO.cleanup()
```

7.2. Resultados

A lo largo de las reuniones mencionadas en el **apartado 7.1.1** se pusieron una serie de objetivos hardware y software.

Respecto al hardware, la finalidad principal era crear un HAT para distintos SBCs donde se pudieran acoplar distintos sensores analógicos y digitales. La **figura 7.4** muestra el resultado obtenido donde se ve el HAT con diversos sensores conectados, cumpliendo de esta manera el objetivo del hardware.

En cuanto al software, había que abordar el caso de los sensores digitales por un lado, y el de los analógicos por otro.

Los sensores digitales se pudieron programar directamente haciendo una serie de archivos en Python. Sin embargo, los analógicos necesitan un ADC con un software que convierta las señales analógicas en digitales. Para lograrlo se programó el PIC18F4455, obteniéndose lecturas analógicas que por falta de tiempo no pudieron ser analizadas correctamente.

El último de los objetivos era poder cumplir con lo acordado respecto a impartir un taller en el colegio. En la **figura 7.5** y **7.6** se muestra como dos de los grupos realizan satisfactoriamente los ejercicios propuestos. En estos ejercicios el objetivo era iluminar un LED RGB, y como se puede apreciar ambos grupos lo consiguieron. En el **código 7.3** y **7.4** se muestra el resultado creado por uno de los equipos para el ejercicio 1 y 2 respectivamente.

Código 7.3: Código del ejercicio 1 hecho durante el taller por uno de los grupos

```
1  # DECLARAR EL PIN QUE VA A SER ACTIVADO
2  S = 7
3  GPIO.setup(S, GPIO.IN, pull_up_down = GPIO.PUD_UP)
4
5  print ("Compruebe el sensor")
6
7  # Esta función comienza cuando se detecta la señal
8  def funcion(null):
9      print("Señal detectada")
10
11 # Cuando se detecta una señal la salida de la función será activada
12 GPIO.add_event_detect(S, GPIO.FALLING, callback=funcion, bouncetime=100)
```

Código 7.4: Código del ejercicio 2 hecho durante el taller por uno de los grupos

```
import OPi.GPIO as GPIO
import time

GPIO.setboard(GPIO.PCPCPLUS)
GPIO.setmode(GPIO.BOARD)

# 1) COMPROBAR EN QUÉ PINES ESTÁ CONECTADO EL LED
R = 31 #PIN para el led Rojo
G = 33 #PIN para el led Verde
B = 35 #PIN para el led Azul

GPIO.setup(R, GPIO.OUT, initial= GPIO.LOW)
GPIO.setup(G, GPIO.OUT, initial= GPIO.LOW)
GPIO.setup(B, GPIO.OUT, initial= GPIO.LOW)

# Bucle principal
try:
    while True:
        print("LED_R_ON_durante_1_segundos")
        GPIO.output(R,GPIO.HIGH) #R ON = R HIGH
        GPIO.output(G,GPIO.LOW) #G OFF = G LOW
        GPIO.output(B,GPIO.LOW) #B OFF = B LOW
        time.sleep(3) # Espera 3 segundos

# 2) HACER QUE G SE ENCIENDA
        GPIO.output(R,GPIO.LOW) #R ON = R HIGH
        GPIO.output(G,GPIO.HIGH) #G OFF = G LOW
        GPIO.output(B,GPIO.LOW) #B OFF = B LOW
        time.sleep(3) # Espera 3 segundos

# 3) HACER QUE DESPUÉS DE ENCENDER R y G SE ENCIENDA B Y ESPERE 3 SEGUNDOS
        GPIO.output(R,GPIO.LOW) #R ON = R HIGH
        GPIO.output(G,GPIO.LOW) #G OFF = G LOW
        GPIO.output(B,GPIO.HIGH) #B OFF = B LOW
        time.sleep(0.5) #



# Fin del programa
except KeyboardInterrupt:
        GPIO.cleanup()
```

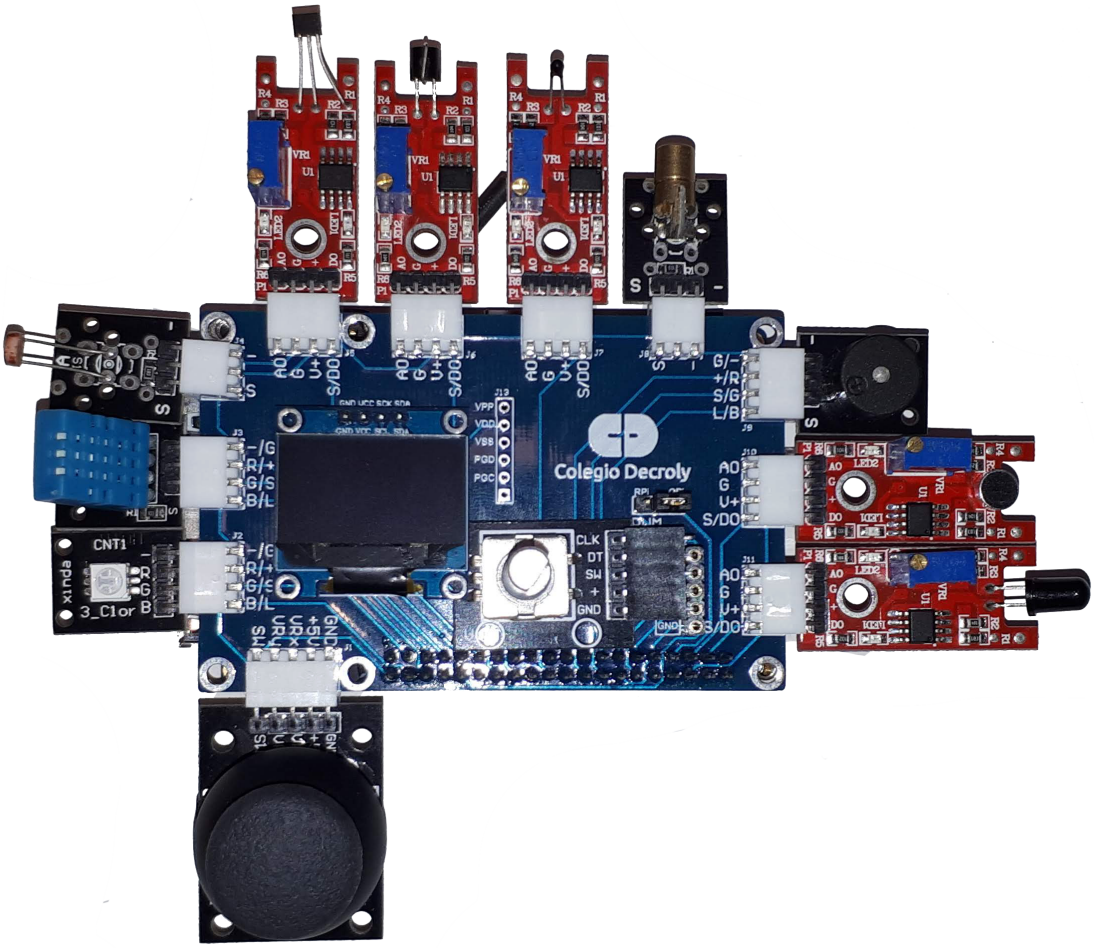


Figura 7.4: Orange Pi con diversos sensores analógicos y digitales acoplados

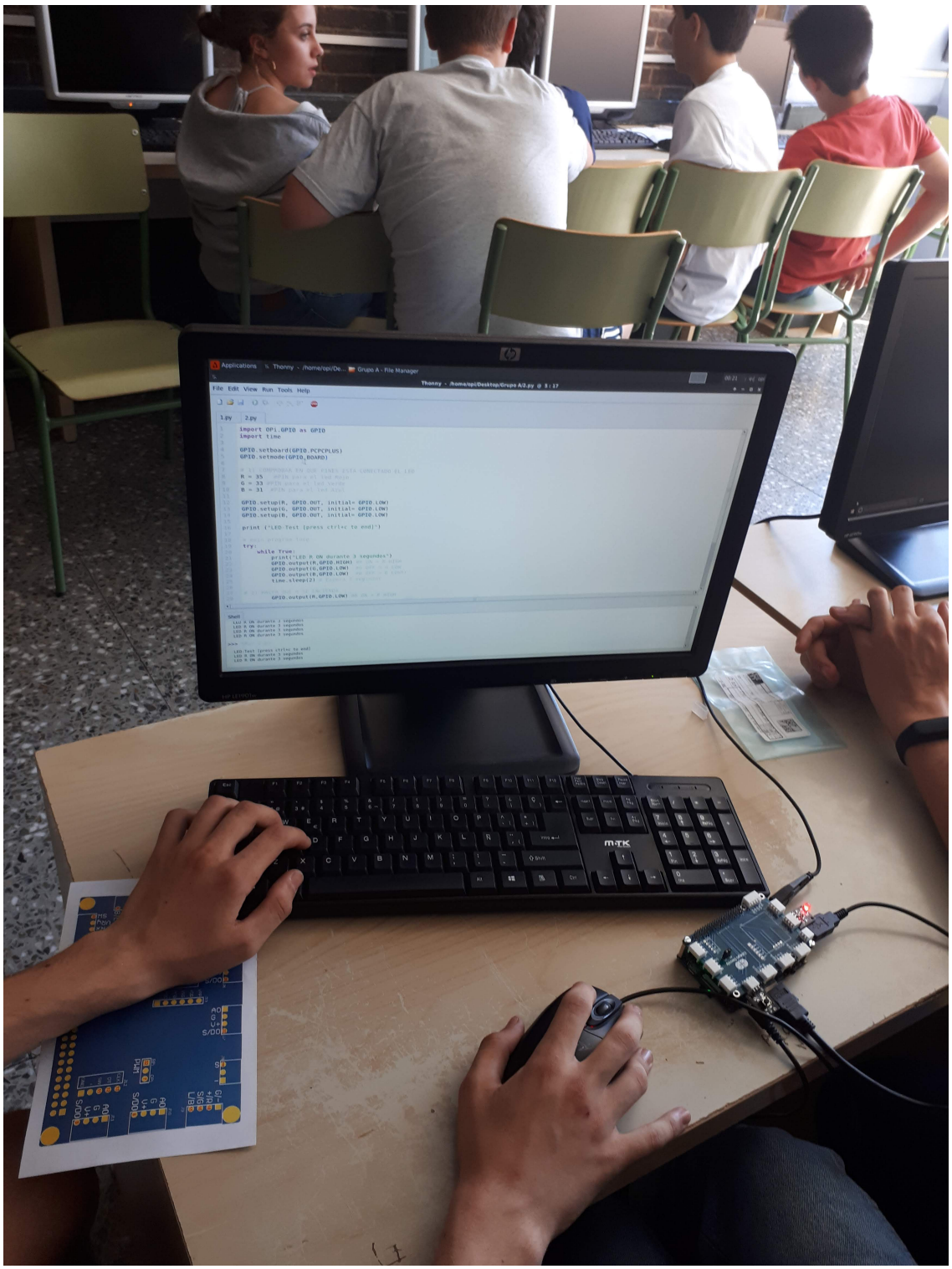


Figura 7.5: Grupo de estudiantes del Colegio Decroly durante el taller programando en Python

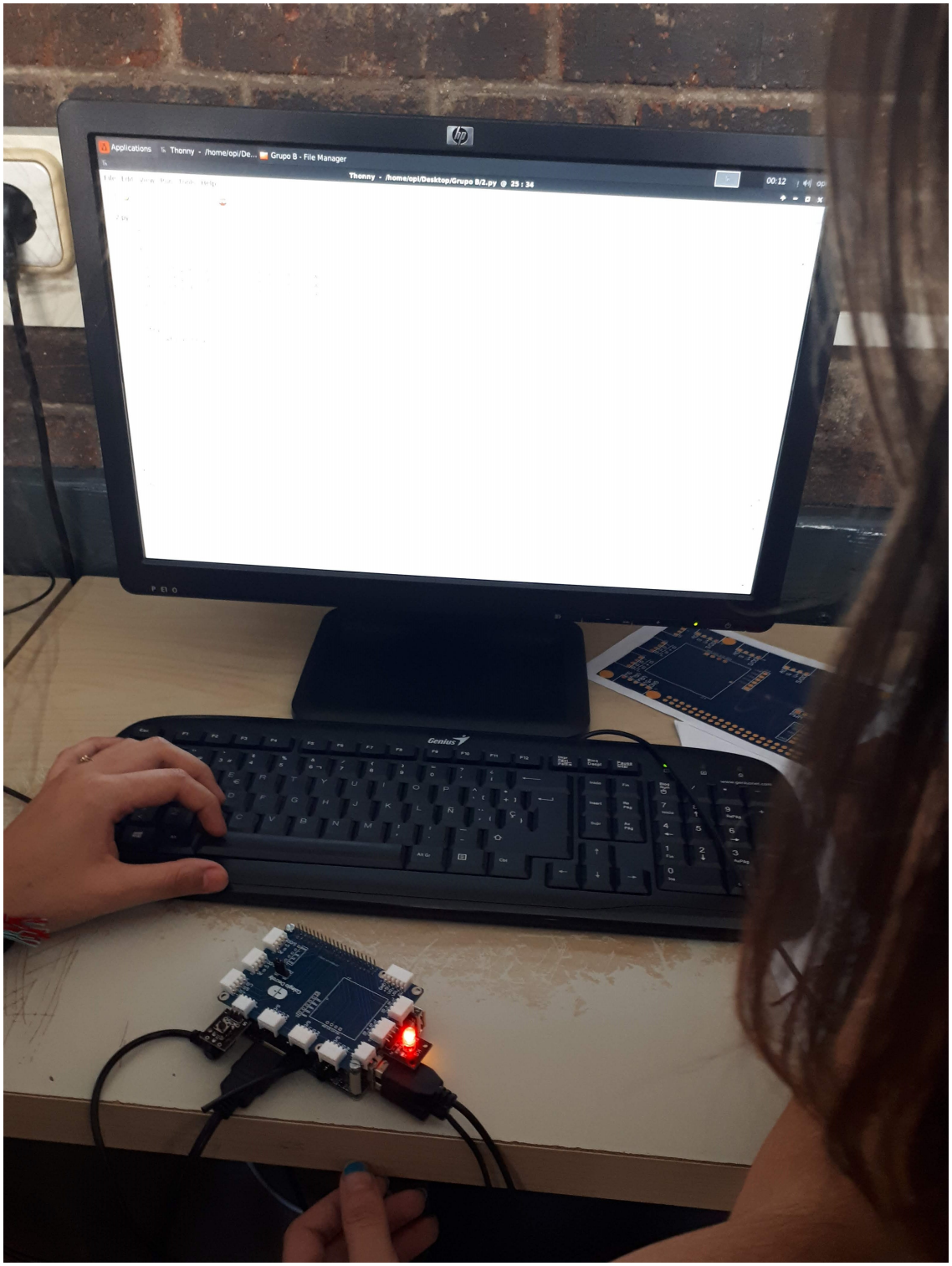


Figura 7.6: Grupo de estudiantes del Colegio Decroly durante el taller consigue iluminar un LED

8

Conclusiones y trabajo futuro

Teniendo en cuenta todo lo mencionado a lo largo de este documento, este Trabajo de Fin de Grado ha cumplido con los objetivos propuestos.

Por un lado, se ha logrado desarrollar satisfactoriamente un HAT que se acople al puerto de 40 pines de distintos SBCs. Además, se ha cumplido un requisito indispensable, ya que para que fuera de bajo coste se pretendía que costase como máximo 50€ el lote completo con todo lo necesario para su uso y finalmente se ha conseguido por 42,51€.

Por otra parte, se ha conseguido impartir un taller en el Colegio Decroly con estudiantes, de forma que el proyecto pudo ser evaluado por los propios estudiantes que hicieron comentarios positivos y se mostraron interesados en lo que estaban aprendiendo.

En cuanto al proceso de concepción de las ideas, la labor de investigación del estado del arte, la definición del sistema que mejor se ajustaba a este proyecto, así como el diseño, desarrollo y experimentación, ha requerido más tiempo del estipulado para un Trabajo de Fin de Grado. Esto ha permitido progresar no solo en el ámbito académico, sino también en el ámbito profesional al tener que realizar un proyecto para un colegio, teniendo que acudir a reuniones, realizar presupuestos y ajustarse a las necesidades del cliente.

Este proyecto ha sido muy útil para involucrarse en la labor real de un ingeniero, además de que aporta en cierto modo mejoras a la sociedad y al sistema educativo actual.

En relación al trabajo futuro, sería interesante crear una página web con ejemplos para programar los sensores, vídeos formativos y complementos que podrían añadirse, como por ejemplo una carcasa impresa en 3D que proteja la placa. También podría crearse el mismo proyecto para un SBC que tanga un tamaño más pequeño y que sea más barato.

Bibliografía

[1] Academias, *10 Ventajas de Los Niños que Deciden Aprender a Programar - Playcode Academy*.

[2] Altium, *Software de diseño de PCB | Innovación para el diseño de PCB | Altium*.

[3] Arduino, *Arduino - Introduction*.

[4] ——, *Interfacing with Other Software*.

[5] Armbian, *Armbian – Linux for ARM development boards*.

[6] Banana Pi, *BPI-R2 GPIO Pin define · banana pi BPI-R2 open source smart router*.

[7] Colegio Decroly, *Bienvenido al Colegio Decroly*.

[8] Coral With Google, *Dev Board | Coral*.

[9] Enciclopedia de Conceptos, *Programación: Concepto, Ejemplos y Programación informática*.

[10] Euractiv, *INFOGRAPHIC: Coding at school — How do EU countries compare?*

[11] Hardware Libre, *¿Qué es una placa SBC?*

[12] JLCPCB, *PCB Prototype & PCB Fabrication Manufacturer - JLCPCB*.

[13] Lego, *Inicio - Mindstorms LEGO.com*.

[14] Luma OLED, *Luma.OLED: Display drivers for SSD1306 / SSD1309 / SSD1322 / SSD1325 / SSD1327 / SSD1331 / SSD1351 / SH1106 — Luma.OLED: Display drivers for SSD1306, SSD1309, SSD1322, SSD1325, SSD1327, SSD1331, SSD1351, SH1106 3.1.0 documentation*.

[15] Mbtechworks, *I2C, SPI, UART Data Communications on the Raspberry Pi*.

[16] MECD, *Programación, robótica y pensamiento computacional en el aula*, tech. rep., Ministerio de Educación y Formación Profesional, 2018.

[17] Microchip, *PIC18F4455 - Microcontrollers and Processors*.

[18] ——, *PICkit™ 3 In-Circuit Debugger*.

[19] MPLAB, *MPLAB X IDE | Microchip Technology*.

[20] Orange Pi, *Orange Pi PC+*, 2016.

[21] Orange Pi Zero, *Orange Pi Zero - Orangepi*.

[22] PCBWay, *Fabricación de PCB - Prototipos de PCB de forma sencilla*.

[23] H. O. Puello, *Ciclo de Vida (Conceptos)*.

[24] PuTTY, *A free SSH and telnet client for Windows*.

[25] Python, *General Python FAQ — Python 3.7.3 documentation*.

[26] Raspberry Pi, *Teach, Learn, and Make with Raspberry Pi – Raspberry Pi*.

[27] Raspberry Pi Zero, *Buy a Raspberry Pi Zero – Raspberry Pi*.

[28] J. Ríos, *Descubriendo la Orange Pi | Introducción, documentación, tutoriales y proyectos de plataformas Orange Pi*.

[29] Rohs Certificate, *Rohs-certificate.com | RoHS Certification | ISO Certification | CE Marking Certification*.

[30] Scratch, *Scratch - Imagine, Program, Share*.

[31] S. Walters, *Scratch GPIO*.

Definiciones

PIC18F4455 Microcontrolador programable de interrupciones.

Acrónimos

ADC Analog to Digital Converter.

BPi Banana Pi.

ESO Educación Secundaria Obligatoria.

FP Formación Profesional.

GPIO General Purpose Input/Output.

HAT Hardware Attached on Top.

I2C Inter-Integrated Circuit.

IDE Integrated Development Environment.

LED Light-Emitting Diode.

OLED Organic Light-Emitting Diode.

OPi Orange Pi.

PC Personal Computer.

PCB Printed Circuit Board.

PWM Pulse-Width Modulation.

RGB Red, Green, Blue.

RoHS Restriction of Hazardous Substances.

RPi Raspberry Pi.

SBC Single Board Computer.

SMD Surface-Mount Device.

SPI Serial Peripheral Interface.

TFG Trabajo de Fin de Grado.

UART Universal Asynchronous Receiver-Transmitter.

Apéndices

A

Guía para utilizar la Orange Pi

Esta guía ha sido creada para dar la posibilidad al usuario de que instale por su cuenta todas las herramientas necesarias para utilizar el PCB en la Orange Pi [28].

A.1. Instalación del sistema operativo Armbian en la Orange Pi

1.– Descargar la imagen del sistema operativo de Armbian desde su web oficial `https://www.armbian.com/orange-pi-pc-plus/`.

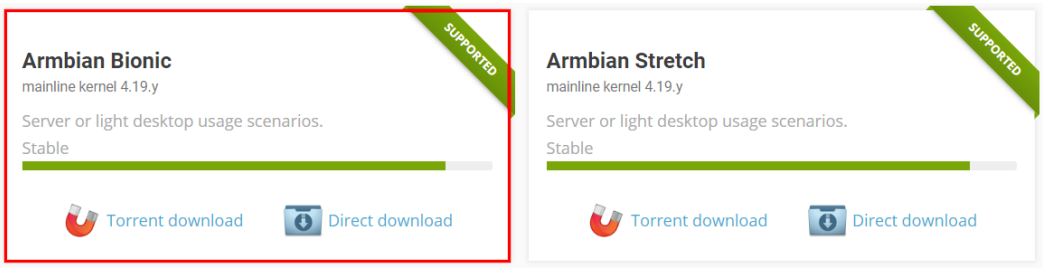


Figura A.1: Sistema operativo Armbian Bionic

2.– Ir a la carpeta donde se haya descargado y descomprimir el archivo con terminación **.img**, que es la imagen que va a ser grabada en la tarjeta SD.

3.– Introducir la SD en el ordenador y abrir administrador de discos.

4.– Una vez abierto hay que buscar la tarjeta SD que ser quiere grabar. Pulsar el botón derecho y Eliminar volumen. . . . Aparecerá un cuadro de texto como en la **figura A.2** en el que hay que pulsar Sí. En caso de que hubiera varias particiones en la tarjeta de memoria, hacerlo con todas.

5.– Descargar e instalar la herramienta Win32dikimager desde el siguiente enlace `https://sourceforge.net/projects/win32diskimager/files/Archive/` para grabar el sistema operativo en la tarjeta SD. Es importante que se haya hecho correctamente el paso 4.

6.– Ejecutar el programa instalado. Saldrá una pantalla como la de la **figura A.3**. Seleccionar la carpeta que está señalada con la flecha de la izquierda. Buscar el archivo de la imagen descomprimido con extensión **.img**, que es la imagen que se va a grabar en la tarjeta SD.

7.– Seleccionar la tarjeta SD, que en la **figura A.3** tiene la etiqueta E:

8.– Hacer clic en el botón *Write* y esperar unos minutos hasta que termine el proceso. Una vez terminado, la tarjeta estará preparada para ser montada en la Orange Pi.

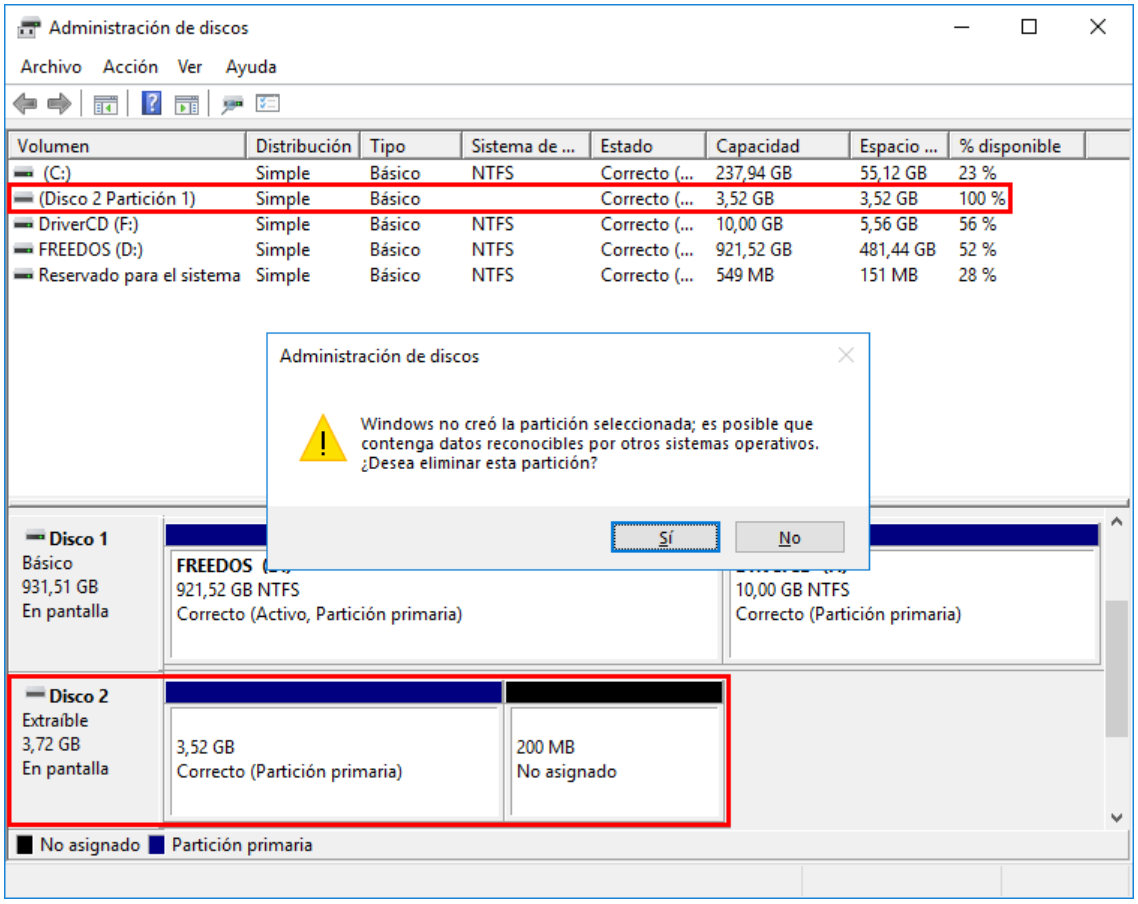


Figura A.2: Administrador de discos en Windows 10

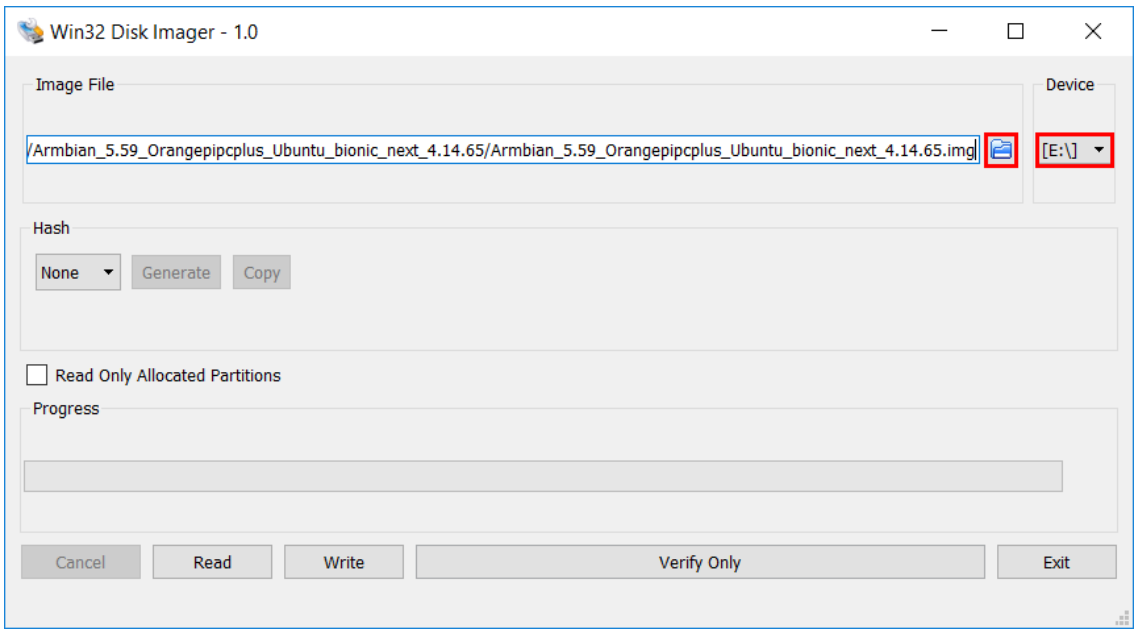


Figura A.3: Win32DiskImager

A.2. Preparación de la Orange Pi por primera vez

Hay dos posibles formas de preparar la Orange Pi para configurarla por primera vez. Se recomienda utilizar el método descrito en el **apéndice A.2.1**, ya que se va a utilizar la Orange Pi con un entorno de escritorio, aunque para la configuración inicial también se puede utilizar el **A.2.2**.

A.2.1. Orange Pi con un entorno de escritorio

Se necesitan los siguientes elementos:

- Orange Pi con su cable de alimentación (2A a 5V).
- Tarjeta MicroSD como mínimo de 4GB.
- Pantalla con su correspondiente cable y, si es necesario, el correspondiente adaptador para convertirlo a HDMI.
- Teclado
- Ratón
- **Opcional** - Cable de red (si no tiene Wifi incorporado)

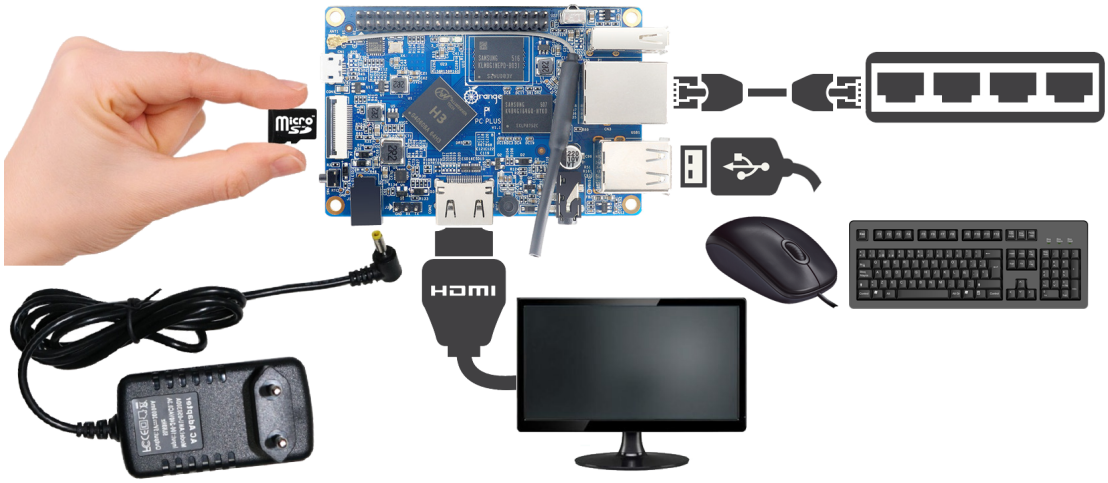


Figura A.4: Esquema de conexión de la Orange Pi

Seguir las conexiones como en la **figura A.4**. Introducir la tarjeta MicroSD en la Orange Pi, conectar el teclado y el ratón a los puertos USB y la pantalla al puerto HDMI. Si se va a conectar a internet mediante un cable de red, conectar un extremo del cable al router y el otro a la Orange Pi. Por último, conectar el cable de alimentación y enchufarlo. Una vez hecho esto se empezará a ver una pantalla negra con muchas líneas. Esperar hasta que solicite que se introduzca el *login* y continuar la guía en el **apéndice A.3**.

A.2.2. Orange Pi como servidor. Conexión con PuTTY y SSH

Para conectarse remótamente, se necesita:

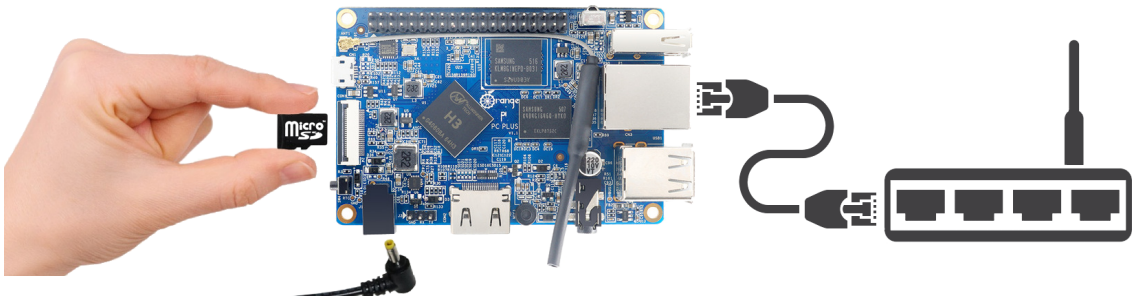


Figura A.5: Esquema de conexión de la Orange Pi

- Orange Pi con su cable de alimentación (2A a 5V).
- Tarjeta MicroSD como mínimo de 4GB.
- Router con acceso a internet.
- Cable ethernet.
- Ordenador.

Introducir la tarjeta MicroSD en la Orange Pi, conectar el cable ethernet por un extremo a la Orange Pi y por el otro al router y, por último, conectar el cable de alimentación a la Orange Pi y enchufarlo, como en la **figura A.5**.

Para el resto de los pasos es necesario un ordenador con conexión a internet, por WiFi o por cable. Es imprescindible que esté conectado al mismo router.

1.– Descargar el programa PuTTY desde el siguiente enlace `https://www.putty.org/` e instalarlo en el ordenador [24].

2.– Entrar en el navegador y acceder al router. Por defecto se puede conectar o bien con la IP 192.168.1.1 o 192.168.0.1, dependiendo del router y la compañía con la que esté contratado el servicio de internet. Introducir el usuario y contraseña del router. En caso de no conocerlos puede encontrarse fácilmente buscando el modelo de router en internet.

3.– Buscar dentro del apartado DHCP la IP que se llame Orange Pi, que deberá de ser del tipo 192.168.1.xxx o 192.168.0.xxx, siendo xxx un número entre 0 y 255.

4.– Abrir PuTTY [24] e introducer la IP de la Orange Pi en el apartado *Host Name (or IP address)* como se puede ver en la **figura A.6**. Pulsar en Open , esperar hasta que soliciten el *login* y continuar la guía en el **apéndice A.3**.

A.3. Configuración en la Orange Pi

A.3.1. Configuración inicial

IMPORTANTE: Después de introducir cada comando hay que pulsar el botón ENTER.

- Introducir login *root* y password *1234*. **Cuando se introduzca la contraseña no se verá lo que se ha escrito, así que después de haberlo escrito pulsar ENTER**.
- En el siguiente paso se va a solicitar el cambio de contraseña. Para ello hay que introducir de nuevo *1234* e

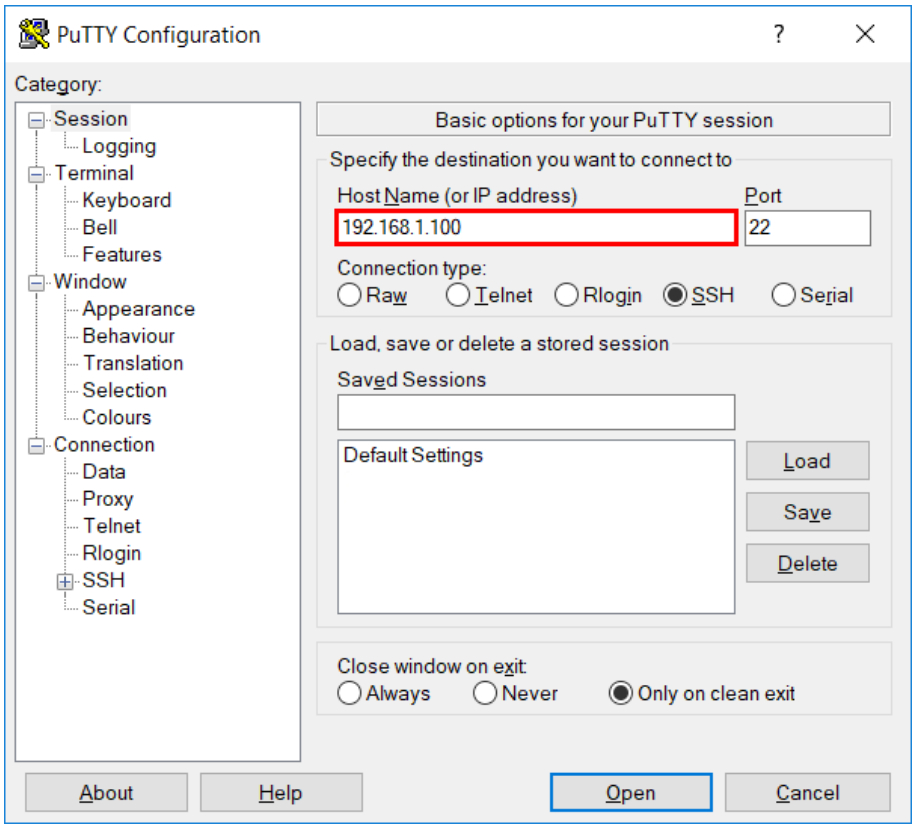


Figura A.6: Menú el programa PuTTY

introducir dos veces la nueva contraseña que le quieras poner. .

- Ahora se va a proceder a crear un nuevo usuario y hay que asignarle una contraseña, que puede ser la misma. También se van a solicitar los datos personales, que pueden dejarse sin rellenar pulsado ENTER.
- A continuación, introducir *armbian-config* (si utilizas el teclado español el botón del guión en lugar de - aparece /, es porque por defecto el sistema utiliza el teclado inglés. Pulsa el botón - en el teclado derecho o el botón de la comilla simple ' o de cerrar interrogación ?) y aparecerá un menú gráfico como el de la **figura A.7**.
- En este menú se pueden usar los cursores para moverse.

Configuración del idioma

Seleccionar *Personal >Locales*. Bajar hasta *es_ES.UTF-8 UTF-8* y seleccionarlo pulsando con el botón Espacio. Aceptar pulsando el botón ENTER y, de nuevo, bajar con el cursor hasta *es_ES.UTF-8* y volver a aceptar.

Si quieres que se efectúe el cambio de idioma hay que reiniciar. Para ello hay que pulsar el botón ESC hasta que salga el terminal de color negro donde habrá que escribir *reboot*. Una vez se haya reiniciado, introducir el usuario *root* y la contraseña nueva contraseña que fue asignada. Para entrar en el mismo menú escribir *armbian-config* y pulsar ENTER.

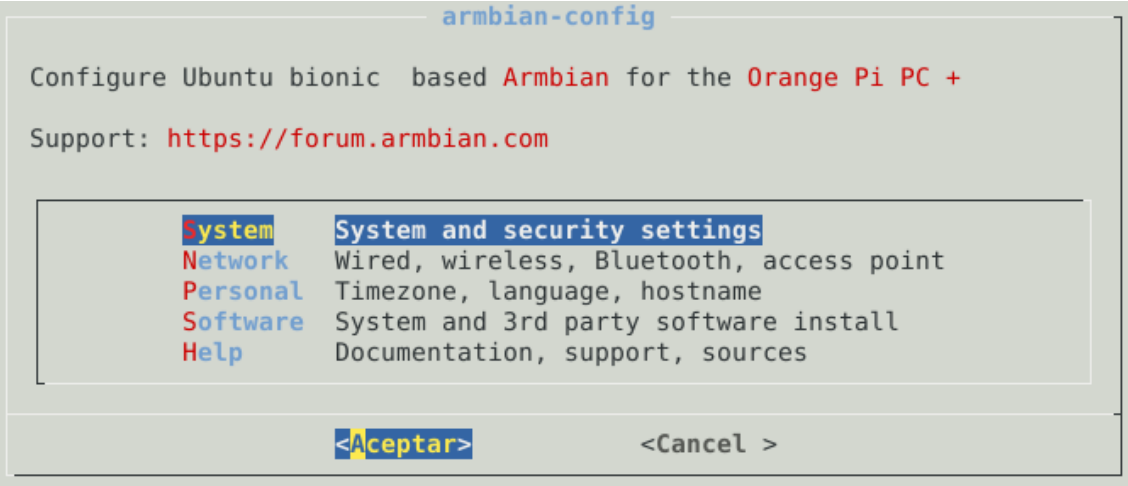


Figura A.7: Menú princial de Armbian

Configuración de teclado

Seleccionar *Personal >Keyboard*. Seleccionar el teclado de la marca correspondiente (si no lo hay seleccionar uno genérico). Bajar hasta el país de origen del teclado (si es España, Spanish). Si el teclado es español y tiene la tecla AltGR a la derecha del Espacio, seleccionar *Spanish*, y en la siguiente pantalla seleccionar *Alt derecho (AltGR)*. Seleccionar como tecla modificadora *Control derecho* y, por último, en utilizar *Control+Alt+Retroceso para terminar el servidor X* seleccionar *<Yes>* y confirmar pulsando ENTER.

Configuración de la zona horaria

Seleccionar *Personal >Timezone*. Seleccionar el continente y una ciudad con el huso horario correspondiente.

Para que se efectúen todos los cambios hay que reiniciar. Para ello hay que pulsar el botón ESC hasta que salga el terminal de color negro donde habrá que escribir *reboot*. Una vez se haya reiniciado, introducir el usuario *root* y la contraseña nueva contraseña que fue asignada. Para entrar en el mismo menú escribir *armbian-config* y pulsar ENTER.

Configuración del WiFi

En caso de que la Orange Pi ya esté conectada por red al router, habrá acceso a internet, pero si no, se puede configurar el WiFi. Entrar en *Network >wlan0 >WiFi*. Seleccionar la red a la que conectarse e introducir la contraseña.

Para que se efectúen todos los cambios hay que reiniciar. Para ello hay que pulsar el botón ESC hasta que salga el terminal de color negro donde habrá que escribir *reboot*. Una vez se haya reiniciado, introducir el usuario *root* y la contraseña nueva contraseña que fue asignada.

Configuración del software

A continuación, se va a proceder a la instalación de varios paquetes poniendo en el terminal los siguientes comandos y pulsando ENTER:

- apt-get install git
- git clone `https://github.com/davidramirezm30/OrangePi-Config-Tutorial`
- cd OrangePi-Config-Tutorial/
- chmod +x install.sh
- ./install-config.sh

Esperar hasta que se actualice todo (podría tardar varios minutos). Reiniciar escribiendo *reboot* y una vez se haya reiniciado, introducir el usuario *root* y la contraseña nueva contraseña que fue asignada.

Escribir *armbian-config* y pulsar ENTER.

- Seleccionar *Software >Full*. Esperar a que se instalen todos los paquetes.
- Seleccionar *Software >RDP*. Se instalará un escritorio remoto al que se puede acceder desde el programa de Windows *Conexión a Escritorio Remoto* conociendo la IP de la Orange Pi.

Para que se efectúen todos los cambios hay que reiniciar. Para ello hay que pulsar el botón ESC hasta que salga el terminal de color negro donde habrá que escribir *reboot*. Una vez se haya reiniciado, introducir el usuario *root* y la contraseña nueva contraseña que fue asignada.

Configuración del sistema

Escribir *armbian-config* y pulsar ENTER. Acceder al apartado System, donde se verá un menú como el de la **figura A.8**.

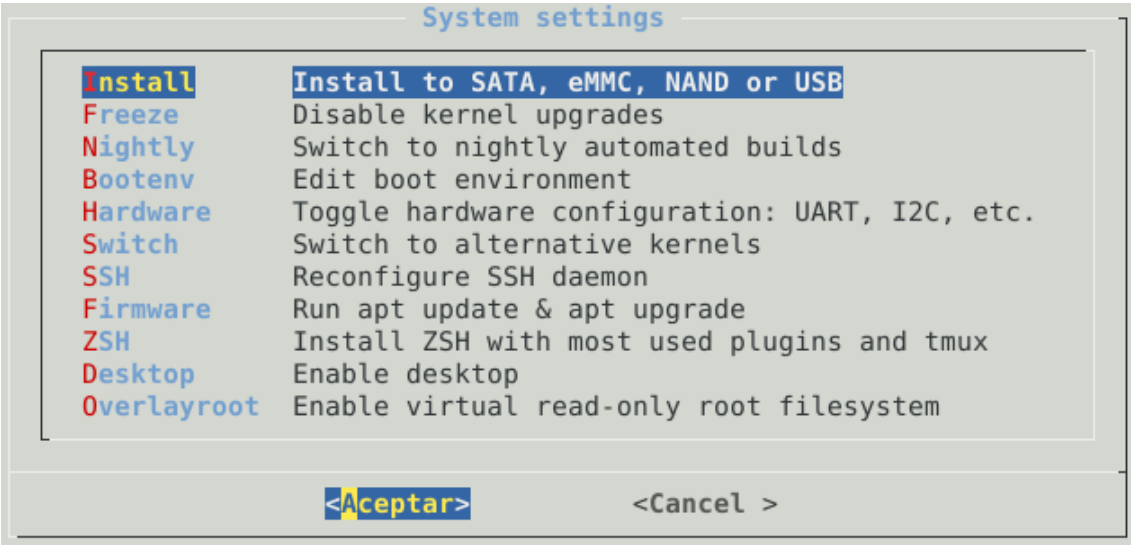


Figura A.8: Menú System de armbian-config

- Seleccionar *System >Firmware*. Esperar hasta que se actualice todo y pulsar ENTER cuando pare para que se reinicie la Orange Pi.
- Seleccionar *System >Hardware*. Seleccionar las siguientes opciones con la tecla Espacio:
    - analog-codec
    - cir
    - i2c1
    - pwm
    - uart3
    - w1-gpio

  Para que se haga efectivo hay que volver a reiniciar pulsando la tecla ESC hasta que salga el terminal de color negro donde habrá que escribir *reboot*. Una vez se haya reiniciado, introducir el usuario *root* y la contraseña nueva contraseña que fue asignada.
- Seleccionar *System >Bootenv* (/boot/armbianEnv.txt) Añadir la siguiente línea para que aparezca como en la imagen de abajo: *dtoverlay=lirc-opi,gpio_in_pin=1,gpio_out=0,gpio_in_pull=up* .
  Para que se efectúen todos los cambios hay que reiniciar. Para ello hay que pulsar el botón ESC hasta que salga el terminal de color negro donde habrá que escribir *reboot*. Una vez se haya reiniciado, introducir el usuario *root* y la contraseña nueva contraseña que fue asignada.
- Seleccionar *System >Default*. Esperar hasta que se instale todo y cuando se reinicie aparecerá el escritorio.

B

Sensores

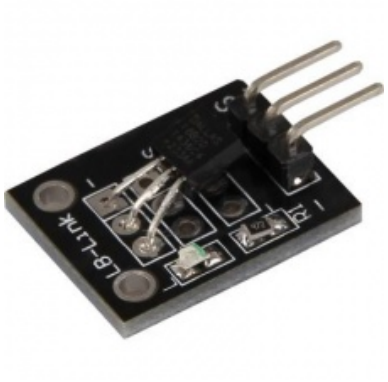


Figura B.1: Módulo sensor de temperatura

Figura B.2: Módulo detector de vibración

Figura B.3: Módulo sensor magnético tipo Hall

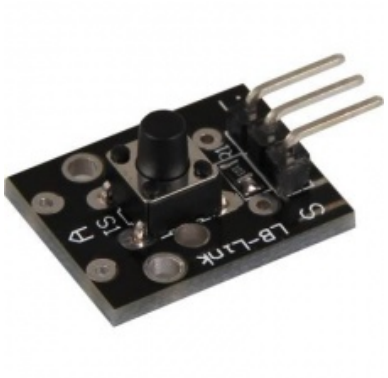


Figura B.4: Módulo interruptor

Figura B.5: Módulo transmisor de infrarrojos

Figura B.6: Módulo sensor de timbre pasivo

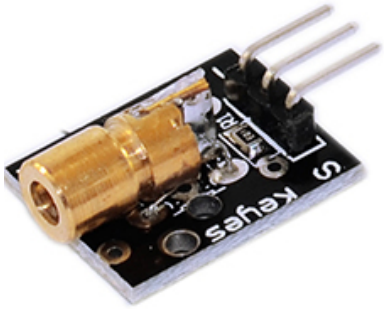


Figura B.7: Módulo láser

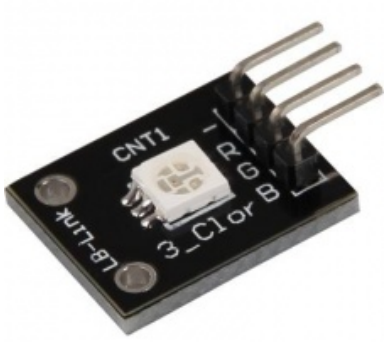


Figura B.8: Módulo LED SMD RGB

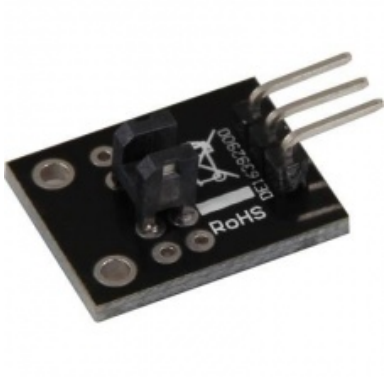


Figura B.9: Módulo de barrera de luz

Figura B.10: Módulo LED de 2 colores

Figura B.11: Módulo sensor de timbre activo

Figura B.12: Módulo sensor de temperatura

Figura B.13: Módulo sensor de temperatura y humedad

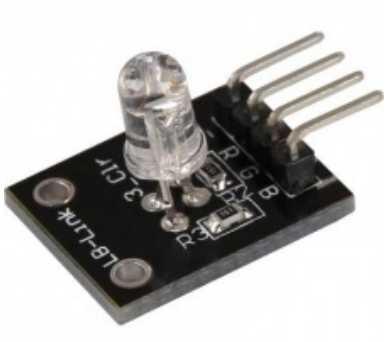


Figura B.14: Módulo LED RGB

Figura B.15: Módulo interruptor de mercurio

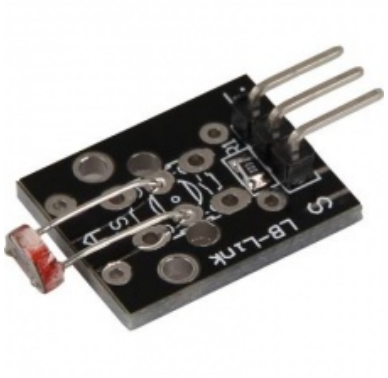


Figura B.16: Módulo fotorresistor

Figura B.17: Módulo relé a 5V

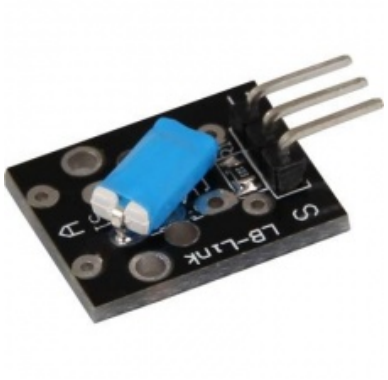


Figura B.18: Módulo sensor de inclinación digital

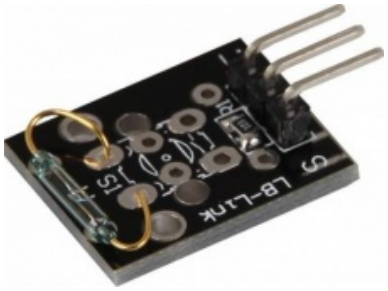


Figura B.19: Módulo magnético de caña

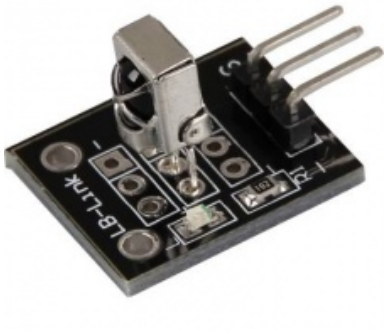


Figura B.20: Módulo sensor receptor de infrarrojos

Figura B.21: Módulo joystick

Figura B.22: Módulo sensor de Hall magnético lineal

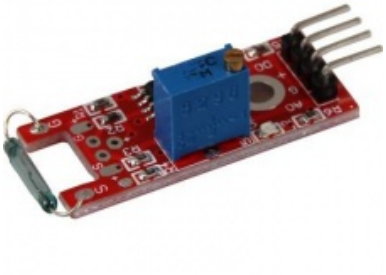


Figura B.23: Módulo magnético de caña

Figura B.24: Módulo sensor de llama

Figura B.25: Módulo interruptor de mercurio con led

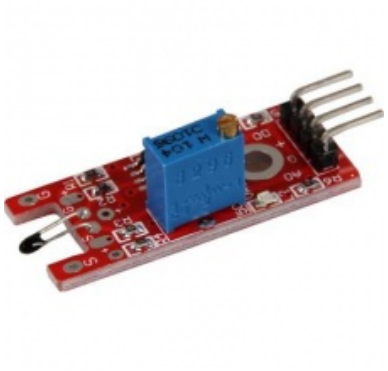


Figura B.26: Módulo sensor de temperatura (termistor)

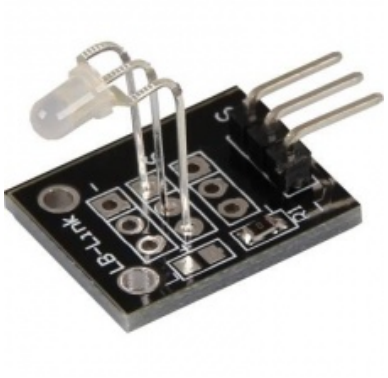


Figura B.27: Módulo LED de 2 colores

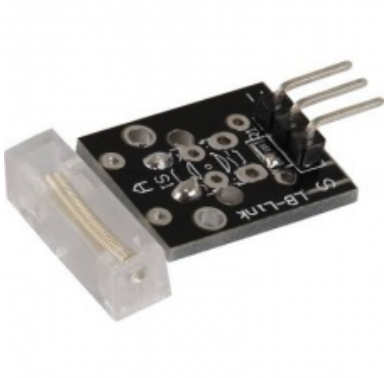


Figura B.28: Módulo sensor de golpe

Figura B.29: Módulo de detección de obstáculos

Figura B.30: Módulo sensor de seguimiento

Figura B.31: Módulo LED de 7 colores parpadeando

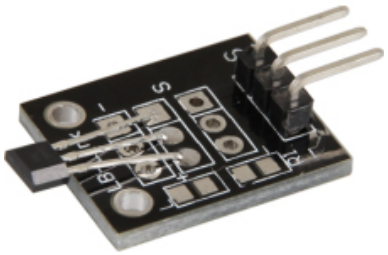


Figura B.32: Módulo sensor magnético Bihor

Figura B.33: Módulo sensor de metal

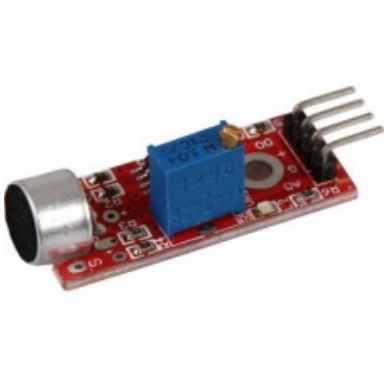


Figura B.34: Módulo micrófono de alta sensibilidad

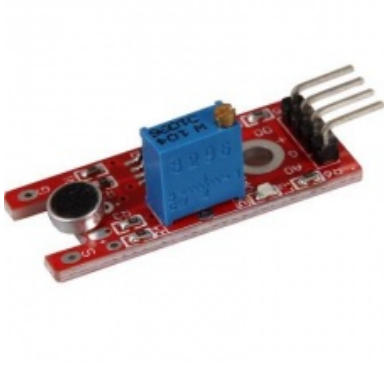


Figura B.35: Módulo micrófono

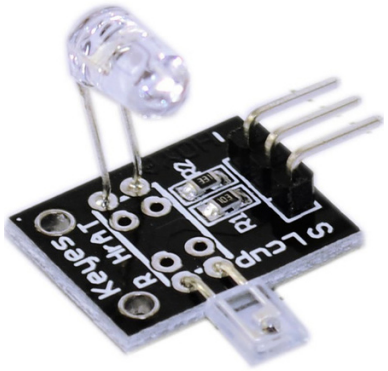


Figura B.36: Módulo sensor de latidos

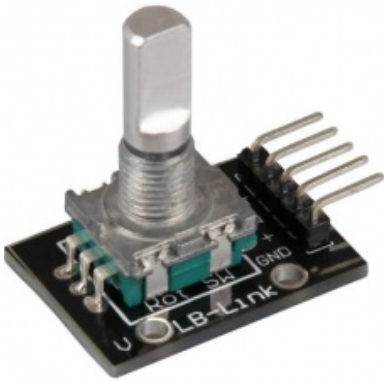


Figura B.37: Módulo de rotación con pulsador